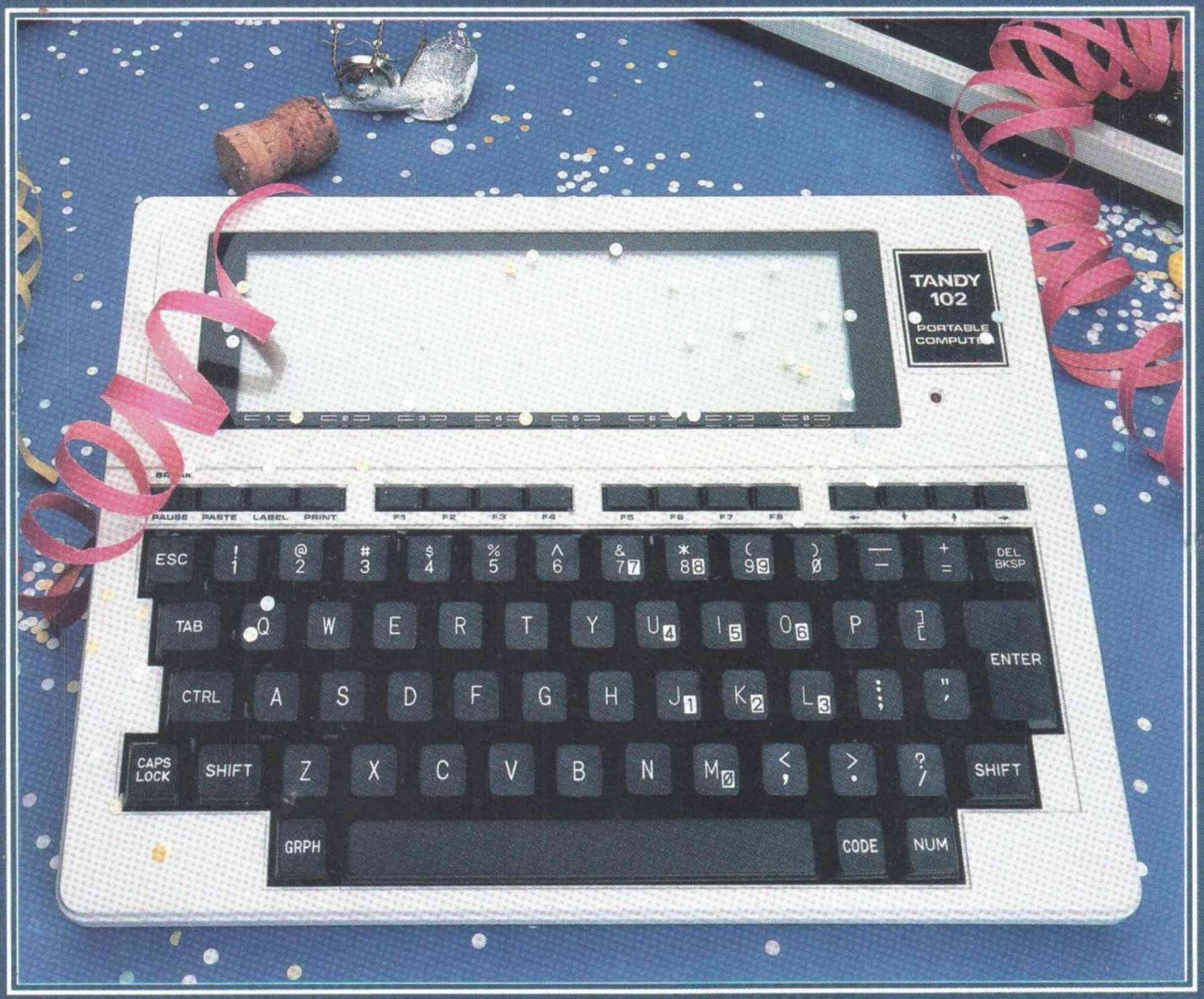
NOW PUBLISHED BY THE WORLD'S LEADING COMPUTER MAGAZINE PUBLISHER!

A CW Communications/Peterborough Publication $3.95

portable 100

TANDY LAPTOP COMPUTING VOLUME 4 NUMBER 1 AUGUST 1987



IN CELEBRATION OF THE TANDY 102

ALSO THIS ISSUE:

Ever Dream of a Faster Modem, Longer Battery Life, and All the RAM Memory and ROM Software you want?

Wake Up! BOOSTER PAK is Here!

Traveling Software introduces the BOOSTER PAK! An incredible breakthrough for Tandy Model 100/102 laptop computers. The BOOSTER PAK gives you an unprecedented 2 megabytes of RAM memory and ROM software. At a very affordable price.

Flexible Design

The BOOSTER PAK is designed to meet your needs now and in the future. You can customize it as you go.

Easy Installation

Just snap the BOOSTER PAK onto the bottom of your Tandy 100/102. Plug in two cables and you're in business.

Built-in Software

Equally as incredible as the hardware is the built-in BOOSTER PAK software. Our unique virtual RAM disk software eliminates any need to do "bank switching." File subdirectories are supported for easy organization of your data. BASIC programs can access files as large as your RAM memory—up to two megabytes! We have also included our popular TS-DOS disk operating system for use with the Tandy disk drive and our new Desk-Link software, as well as an XMODEM program to upload/download files up to 2 megabytes directly to the virtual RAM disk. And to top it off, we have even included a fast action Asteroids game with amazing graphics in our standard BOOSTER PAK.

The BOOSTER PAK Everything You Asked For and More!

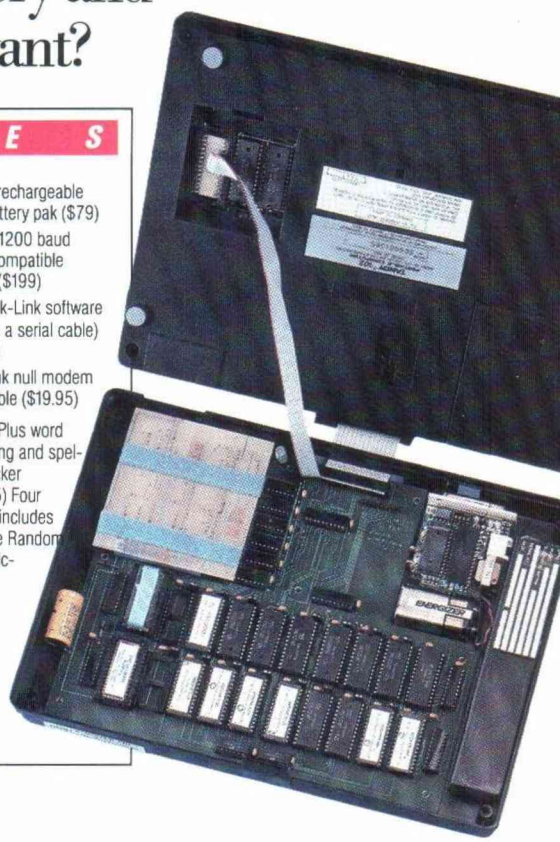
F E A T U R E S

Here's what you get in a standard BOOSTER PAK for only $429:

- 128K of RAM memory (96K available for files)
- 64K of ROM software
- 16 socketed slots for 32K RAM and/or ROM software chips
- 2 Tandy Molex ROM sockets
- Virtual RAM software (no bank switching or copying)
- File folders/sub-directories supported
- TS-DOS disk software built-in
- XMODEM file upload/downloading to virtual RAM
- Long-life lithium battery for RAM backup (rated at 7 years)
- BASIC programming access to virtual RAM
- ROM environment software (mix and match any ROM software)
- Ability to develop ROM software from 32-480K in size
- Fast action Asteroids game (with great graphics) built-in

BOOSTER PAK Options

- 32K RAM (certified low power) chips ($20 ea.)
- 6 slot RAM expansion board ($68)
- 256K RAM expansion modules (requires expansion board) ($159 ea.)
- Internal rechargeable nicad battery pak ($79)
- Internal 1200 baud Hayes-compatible modem ($199)
- IBM Desk-Link software (requires a serial cable) ($39.95)
- Desk-Link null modem serial cable ($19.95)
- Sardine Plus word processing and spelling checker ($199.95) Four chip set includes complete Random House dictionary



Booster Pak™

*To order **BOOSTER PAK** call toll free*

1 800 343-8080

Or write: **Traveling Softwa**

TRAVELING SOFTWARE

North Creek Corporate Cent
19310 North Creek Parkway
Bothell, Washington 98011
206-483-8088

What do you call thousands of words packed into one powerful chip?

The first electronic pocket dictionary for laptop computers.

If you have a laptop computer and you don't have **Sardine**, you're missing out on one of the most powerful laptop tools available today.

It's **Sardine**. The *first* laptop pocket dictionary. The *first* laptop spelling checker. The *first* to be put on a plug-in ROM chip. And the *first* to give you access to 33,000 words from what has been called "the best vestpocket dictionary in the world"—the Random House American dictionary. *(You even get your own printed copy of the dictionary with every* ***Sardine*** *you buy).*

Faster than a desktop!

With **Sardine** and virtually any word processing software, you can look up a single word in seconds or check an entire document up to 25K in size in less than three minutes. All you do is press two keys while you're typing and you can check just the word you're working on or your entire document.

Random House and Wang Labs quality.

Sardine gives you so much more because it gives your laptop computer capabilities you may never have imagined. In addition to the 33,000 word Random House™ American dictionary, over *7,000* of the most frequently used business words are included in the ROM version. These priority words are the results of a two-year, one million word study of business correspondence conducted by Wang Laboratories' lexicographers.

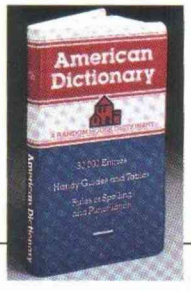
And that's not all. You can also word process with **Sardine**, because it contains the acclaimed T-word word processing program.

Able to create custom dictionaries with a single key.

Just press one key and **Sardine** lets you create an auxiliary, custom dictionary to which you can add words, names, abbreviations or any special terms you use. Add hundreds, even thousands of words, depending on available memory.

F E A T U R E S

Sardine is available for the Tandy Models 100/102, 200, and NEC PC-8201 laptop computers.
Requirements: 16K RAM, Tandy/Purple Computing disk drive optional for ROM version.



- **Sardine** is available on both a 3½ inch disk and a plug-in ROM chip.
- Super fast! Processes over 13,000 dictionary words a minute.
- Look up a single word in seconds. Or, check an entire document up to 25K in less than three minutes. Corrections are automatically inserted.
- ROM version contains the acclaimed T-word word processing program and over 7,000 "priority" words on the ROM chip itself. (ROM version also works with the disk).
- Create your own auxiliary dictionary for special terminology.

P R I C I N G

3½" Disk version $99.95

Plug-in ROM version $169.95
(includes T-Word word processing)

Circle 77 on Reader Service Card

SARDINE

To order ***Sardine*** *call toll free:*

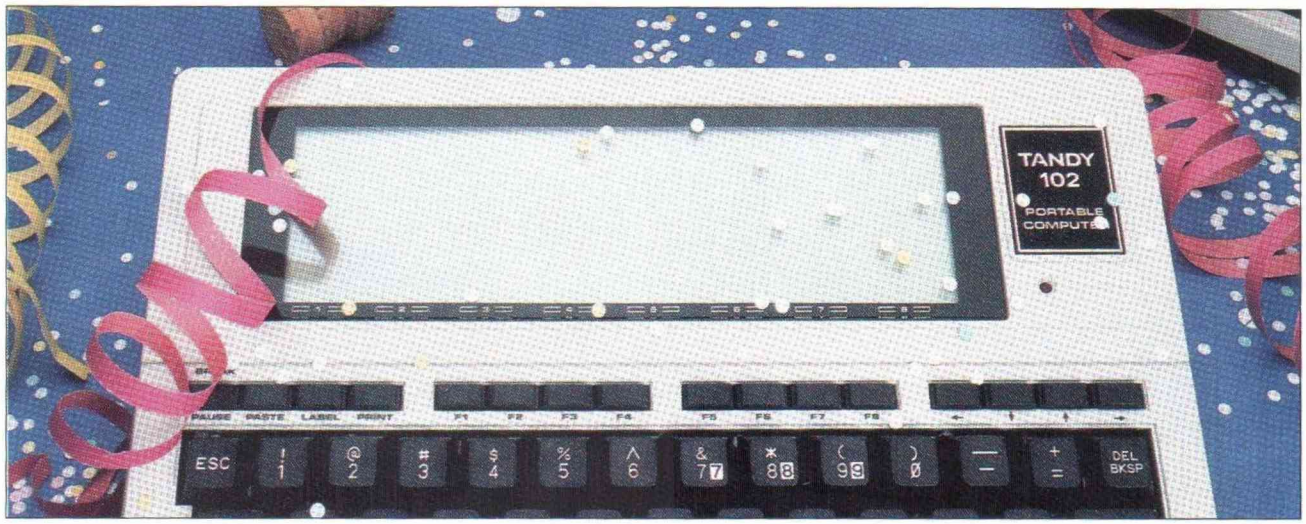
1 800 343-8080

Or write: **Traveling Software**
North Creek Corporate Center
19310 North Creek Parkway
Bothell, Washington 98011
206-483-8088

TRAVELING SOFTWARE

portable 100

AUGUST 1987 VOLUME FOUR, NUMBER ONE

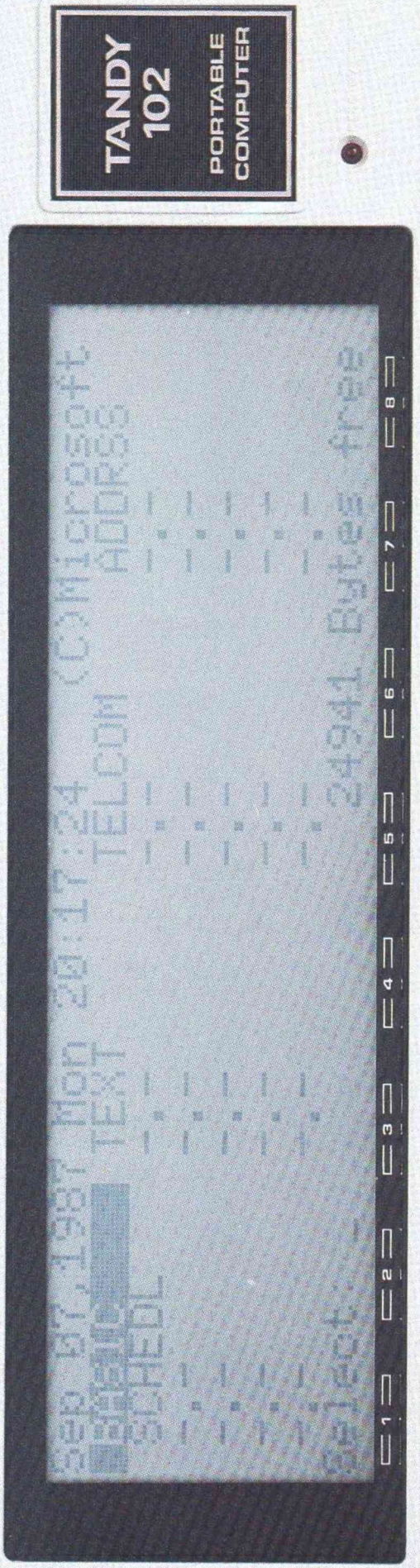


ON THE COVER

FEATURES

DEPARTMENTS

Cover photo by Benjamin Magro



The Tandy® 102
Try one on for size.

Don't let the size fool you. If you had a Tandy 102 portable computer on your lap you'd have your hands on a lot of power.

Fact one: The Tandy 102 comes with five built-in programs. Use it as a personal word processor, an address/phone directory, appointment calendar and auto dialer. Or write your own programs in BASIC.

Fact two: A built-in modem lets you communicate with other computers over the phone. Keep in touch on the road. Or access national information services.

Fact three: You get all this for just $499. The Tandy 102 packs the power of a desktop computer in a small, battery-powered package that weighs only three pounds.

Come into Radio Shack today and try on a Tandy 102 (26-3803) for yourself.

Radio Shack®
The Technology Store™
A DIVISION OF TANDY CORPORATION

Price applies at Radio Shack Computer Centers and participating stores and dealers. Actual size: 11½ × 11⅞ × 8½". Batteries not included

Circle 42 on Reader Service Card

Classic Revival

It is my pleasure to announce the purchase of Portable 100 magazine by CW Communications/Peterborough. CW/P is part of IDG Communications, the world's largest publisher of computer magazines and information. IDG publications include Computerworld, PC World, MacWorld, 80 Micro, InCider, Run and AmigaWorld magazines. (See page 58 for a full listing of IDG activities.) IDG and particularly CW/Peterborough have staked a claim as the number one publisher of magazines dedicated to specific computers. Recently, CW/P has introduced publications on leading-edge topics, magazines like CD-ROM Review and Portable Computer Review.

Portable 100 fits into this publishing environment quite nicely. Dedicated to a single computer family and committed to the creative, leading-edge applications its readers have discovered, Portable 100 has built up a loyal audience of readers and advertisers since its inception in 1984. Now, with IDG behind it, its future's so bright, it's gotta wear shades.

The excitement starts with this issue. My first responsibility as editor is to serve the reader. To meet that goal, I've taken the best of the old Portable 100 and brought it to the new. Technical Editor Alan Zeichick remains on the masthead, and is relocating to CW/P's Peterborough, New Hampshire office. Long-time Contributing Editor Carl Oppedahl will continue to be featured regularly in the magazine's pages. The wonderful cartoons of Andi McBrian will remain. In fact, I've made only the slightest changes to the design — Portable 100's readers have always been happy with the editorial product, and I see no reason to mess that up.

Down the road, look for some new columnists, new feature writers, and improved reviews and product listings. CW/Peterborough has bought a good product, but I think we can make it even better. We will be publishing on a monthly basis, and subscribers will get all the issues owed to them.

I'm going to keep the editorial short this month, to let you get right to the heart of the magazine. In this issue, you'll find:

- two features on the Tandy 102
- programming articles describing how to use your computer to track the sun, create a spreadsheet, and personalize your form letters.
- a programming workshop
- a how-to article about building a low-cost analog-to-digital converter
- and like they say on the TV commercial, "there's more." Much more.

Let me know what you think. Write to me at 80 Elm St., Peterborough, NH 03458.

Happy reading! — Roger Strukhoff, Editor

Editor
Roger Strukhoff
(603) 924-9471

Managing Editor
Steven F. Smith

Technical Editor
Alan L. Zeichick

Assistant Editor
Janet M. Tiampo

Senior Contributing Editor
Carl Oppedahl

Contributing Editors
G.N. Gilbert
Louis C. Graue
Kevin Jones

Design Consultant
Peter D. Koons
High Resolution

Publisher
Jeff DeTray

Associate Publisher
Jim McBrian
(603) 924-9471

Advertising Representative
Carl Cramer
Vertical Marketing Group — (207) 594-6216

Advertising Coordinator
Michelle Boot

Corporate Production Director
Dennis Christensen

Manufacturing Manager
Susan Gross

Assistant Manufacturing Manager
Leslie Walden

President/CEO
Michael S. Perlis

Vice President/Finance
Roger Murphy

Director of Circulation
Frank S. Smith

Circulation Manager
Bonnie Welsh

Single Copy Sales Manager
Linda Ruth

Direct Marketing Manager
Paul Reuss

Telemarketing Manager
Elizabeth R. Kehn

Director of Credit Sales
William M. Boyer

ISSN 0893-942X

Portable 100 is an independent journal not connected with any hardware, software, or peripheral equipment manufacturer. Portable 100 is published by CW Communications/Peterborough, Inc., 80 Elm St., Peterborough, NH 03458. Second Class postage paid at Peterborough, NH and additional mailing offices. Subscription rates in the U.S. are $24.97 for 12 issues. In Canada, $45.97 (Canadian). In Mexico, $29.97, U.S. funds drawn on a U.S. bank. Other foreign subscriptions (surface mail) are $44.97 for 12 issues; air mail is $79.97, U.S. funds drawn on a U.S. bank. Address all subscription correspondence to Portable 100, Subscription Dept., 80 Elm St., Peterborough, NH 03458. POSTMASTER: Send address changes to Portable 100, 80 Elm St., Peterborough, NH 03458. *Volume 4, Number 1, August, 1987.*

SOUNDSIGHT ANNOUNCES

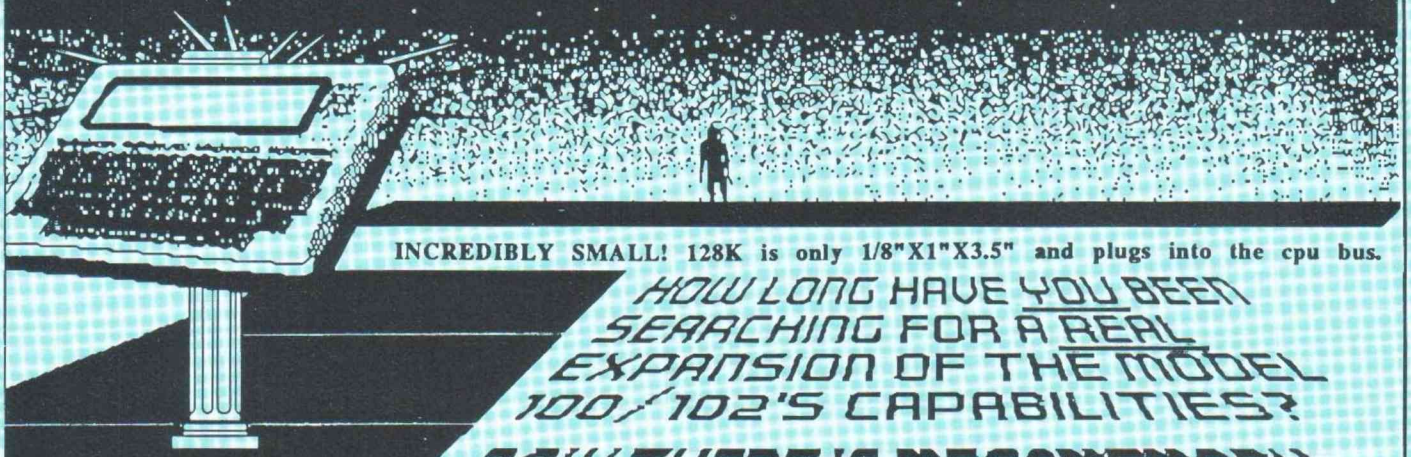
MEGAMEMORY

A NEW ERA IN RAM.

128K TO 10 MEGABYTES

RAM EXPANSION FOR THE TANDY 100 AND 102 COMPUTERS

CREATES AND EDITS* FILES AS LARGE AS THE AMOUNT OF RAM PURCHASED. FULL I O FUNCTIONS IN BASIC. FASTER, IN SOME APPS, THAN THE MODEL 100/102. OPTIONAL 4 PLACE ROM BANK. PARALLEL CPU BUS. TOTAL SIZE OF 1 MEGABYTE MODULE IS 1"X1.3"X3.5". DOES NOT COVER COMPUTER'S I O PORTS OR BATTERY COMPARTMENTS.



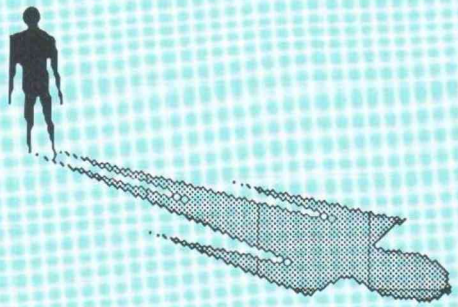
INCREDIBLY SMALL! 128K is only 1/8"X1"X3.5" and plugs into the cpu bus.

HOW LONG HAVE YOU BEEN SEARCHING FOR A REAL EXPANSION OF THE MODEL 100/102'S CAPABILITIES?

NOW THERE IS MEGAMEMORY.

Utilizing the same proven operating system software SoundSight placed in Standard Oil Service Stations from Alaska to Florida, MEGAMEMORY now offers FULL TEXT EDITING of files of any size existing in the RAM bank. *Text Editor Priced at $75.00

Currently undergoing tests for use by the LA Co. Sheriff's Dept, Megamemory also offers 3rd Party Development,(C language), POS systems and Vertical Market Applications.



The advantage of our widely used operating system lies not only in providing the first Text Editor accomodating model 100/102 files greater than 32k, but in our coming support software which includes the Text Editor, Spreadsheet, and Database.

PRICES: 128K-$329.00, 256K-$529.00, 384K-$625.00, 512K-$699.00, 640K-$769.00, 768K-$839.00, 896K-$899.00, 1 MEGABYTE-$965.00.

Additional memory prices available up to 10 megabytes in bottom mounted RAM pan. All RAM bank modules are actual byte sizes listed and do not include the Model 100 or 102's RAM.

CONTACT:

SOUNDSIGHT MBM INC. 225 W. Broadway, Suite 509, Glendale, CA., 91204 Ph (818) 240-8400

Circle 76 on Reader Service Card


TANDY
102
PORTABLE
COMPUTER
PAUSE PASTE LABEL PRINT
F1 F2 F3 F4 F5 F6 F7 F8
ESC
TAB
CTRL
CAPS LOCK
SHIFT
GRPH
CODE
NUM
DEL BKSP
ENTER

Heir to the Throne

The Model 100 is dead; long live the Tandy 102.

By Alan Zeichick and Carl Oppedahl

Smaller, lighter, less expensive than the Model 100; that's the new Tandy 102, Tandy Corp.'s newest portable computer. But how does this new laptop compare to the old, standard-setting Model 100?

Introduced quietly, with little fantle different than its famous father. Like many children, the young laptop is slightly smaller and lighter than the Model 100, which debuted in 1983. The vital statistics: 11.5 by 8.5 by 1.25 inches fare, in the fall of 1986, the Tandy 102 for appears litthe Tandy 102, 11.5 by 8.5 by 1.5 inches for the Model 100. That quarter-inch of height may not seem like much to be excited about — but when your briefcase is bulging at the hinges, every millimeter counts.

The familiar off-white and matte black plastic case is identical, with a few slight exceptions. The keyboard hasn't changed much, except for a rearrangement of the cursor and function keys. The Model 100's eight programmable function keys occupy the two left banks at the top of the keyboard; on the Tandy 102, these eight keys are at top center. According to Tandy sources, this arrangement "makes the 102's keyboard more like a desktop computer's keyboard."

The group of four cursor, or arrow, keys still occupy the top right section of the keyboard — but have been inexplicably reordered, from the 100's *up-down-left-right* to *left-down-up-right*. Why? Nobody seems to know, but unless you intend to be switching from a 100 to a 102 and back again, you'll never notice the difference. Another minor difference between the two machines: The simple slide switches used for the laptop's power, selecting *DIR/ACP* and *ANS/ORIG* are flatter and wider on the Tandy 102.

The Model 100's keyboard was one of the industry's finest, with the right combination of tactile response and key placement. The 102's is very similar, only with a more "metallic" feel reminiscent of the Tandy 200. The keys are all slightly concave. And the easy-to-overlook *Num* (number pad selection) key still threatens to give you *2533* when you meant to type *KILL*.

Another physical change between the Model 100 and the Tandy 102 is the location of the system bus. On the Model 100, the bus connector was tucked into the option read-only memory (ROM) compartment underneath the computer; like the Tandy 200, the Tandy 102 has the bus connector on the back panel, between the printer connector and the reset button. The number and function of the pins is unchanged.

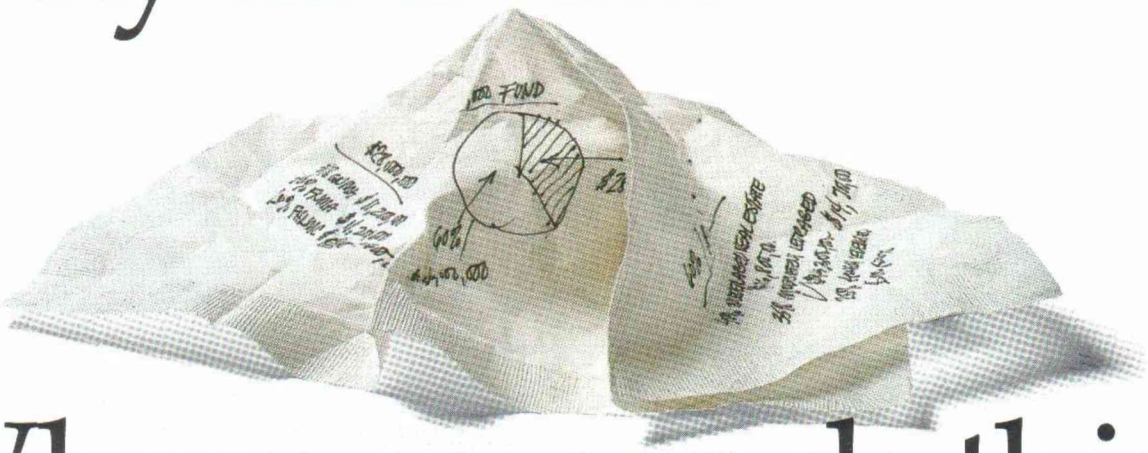
In some regards, the new location of the system bus connector is an improvement; the Model 100's connector is tough on cables, as anyone who owns Tandy's Disk/Video Interface can attest to. Unfortunately, add-on random access memory (RAM) expansion modules for the Model 100 cannot be used as-is on the Tandy 102. Companies such as Portable Computer Support Group, Traveling Software, and PG Designs have had to redesign their printed circuit boards and other hardware add-ons to accommodate the change.

Another hardware improvement is the Tandy 102's RS-232C connector. The Model 100's case left insufficient room for some of the thicker molded serial cables, requiring that some users file down their 100. The Tandy 102's socket has more clearance, making it easier to use off-the-shelf serial cables.

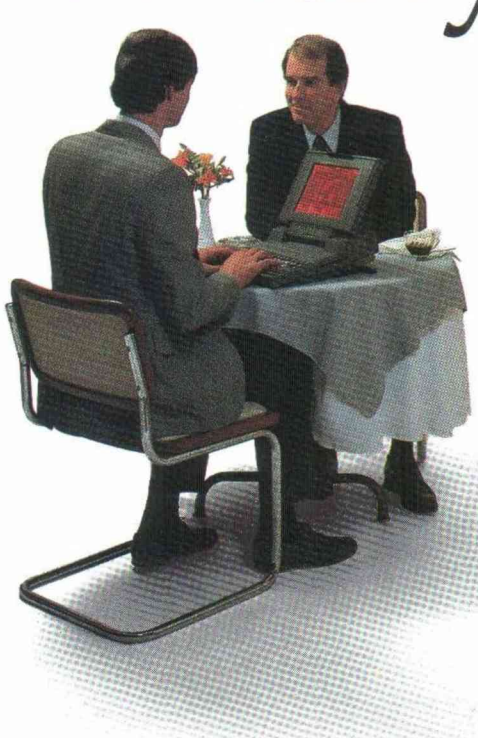
A final noticeable change: The black nameplate next to the LCD reads "Tandy 102 Portable Computer," not "Radio Shack TRS-80 Model 100 Portable Computer." The Radio Shack name and TRS-80 trademark are missing

PHOTOGRAPH BY BENJAMIN MAGRO

Why do this...



When you can do this?



Powerful portable computers can revolutionize the way you do business outside the office environment.

And ***Portable Computer Review*** is the definitive reference source for choosing the portable computer to best suit your needs.

Laptops, transportables, and hand-helds all receive in-depth scrutiny in ***Portable Computer Review***. In addition, the Buyers' Guide is the most complete summary of all the features of all the models on the market.

The substantial range of topics includes...

- what to look for—and what to avoid—when choosing a screen
- how much memory is "enough"
- when is weight important...when is it not
- is compatibility critical
- what to check when testing the keyboard
- how to communicate with your office computer
- how much they cost...and much more!

Make sure you consult ***Portable Computer Review*** before making your purchase decision! Supplies are limited. Order your copy from **CW Communications**—the world's largest publisher of computer magazines—TODAY! For immediate service call toll free 1-800-258-5473.

Yes! I want to make the right decision. Please send me _____ copy(ies) of ***Portable Computer Review*** at $3.95 each.

☐ Payment Enclosed

☐ VISA ☐ MasterCard

☐ American Express

CARD # _____

EXP. DATE _____

SIGNATURE _____

NAME _____

STREET ADDRESS _____

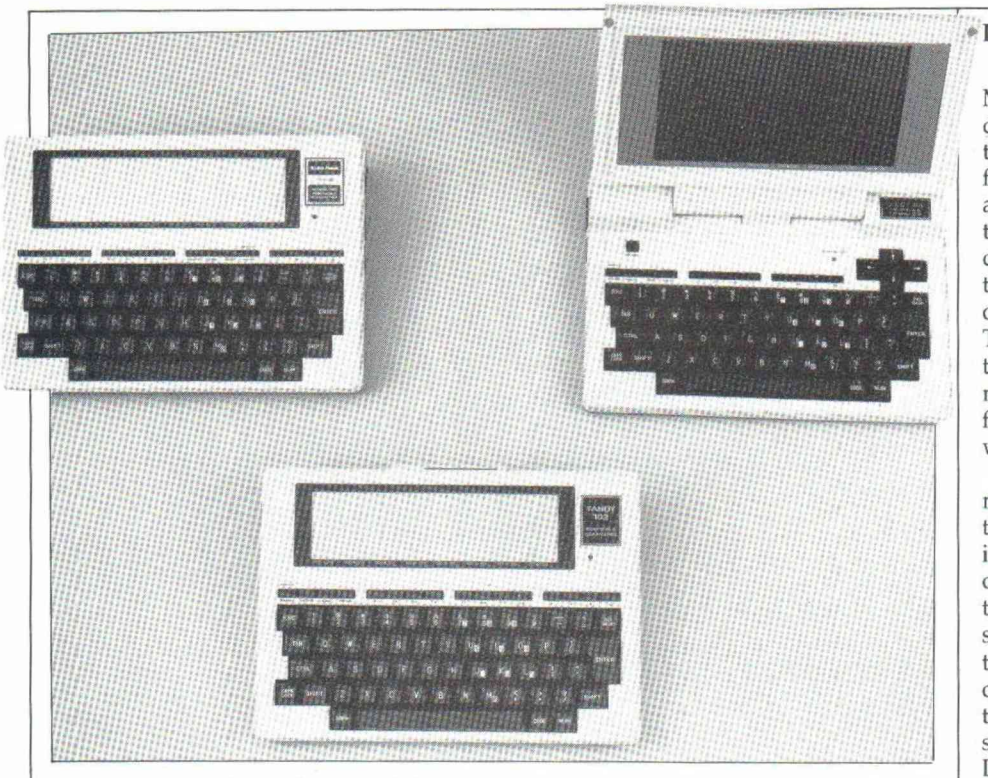
CITY _____ STATE _____ ZIP _____

TELEPHONE _____

3707CD

Please make checks payable to ***Portable Computer Review.***
Delivery will begin mid-March, 1987.

Mail to: ***Portable Computer Review,*** CW Communications/Peterborough
80 Elm Street, Peterborough, N.H. 03458



The Tandy 102's function keys follow the Tandy 200's pattern; however, the cursor keys have established a standard of their own.

from Tandy Corp.'s newer computers, as the company tries to contend with the "local battery store" image. Very few Tandy computers still bear the TRS-80 logo, among them the company's oldest MS-DOS computer, the Intel 80186-based TRS-80 Model 2000.

A LONGER MEMORY

The original Model 100 was shipped with eight kilobytes (K) of RAM, expandable to a maximum of 32K. The Tandy 102's internal maximum is still 32K — but the machine now ships with 24K installed. The built-in socket for adding the final 8K is in the option ROM compartment on the bottom (again like the Tandy 200) rather than inside the case of the computer where it was found on the Model 100. The socket for the last 8K is labeled *M6*, by the way.

This RAM socket does *not* use the same RAM expansion chip as does the Model 100. In the new computer, Tandy has conformed to what's known in chip-manufacturing circles as the *JDEK* standard, instead of the unusual configuration used for memory chips in the Model 100. Several manufacturers make this chips suitable for Tandy 102 RAM expansion; it's no longer Tandy-specific. This is good news for the consumer because it means lower prices. For example, Purple Computing's 8K RAM upgrade for the Model 100 sells for $29 — and for the 102, 8K goes for $5.

Again sticking to its newer Tandy 200 style, the 102 comes with several smaller manuals, instead of the large, metal spiral bound Model 100 book. There's little diffence in the manual content; the Model 100 manual is harder to misplace, but the Tandy 102 manuals are easier to carry around. Fortunately, the Model 100 family is straightforward to operate — and the manuals are used more as quick-reference guides than as tutorials.

The 102 manuals have several quirks and misprints. On page 136 of the *Owner's Manual* and on page 92 of the *Reference Guide* a mysterious "answerback" routine is mentioned. In telecommunications terminology, an *answerback* is an identifying code that a terminal automatically transmits upon receipt of a certain escape sequence, Esc-I. This answerback is used by many larger minicomputers and mainframe to configure a terminal session to meet the communications requirements of a terminal. Unfortunately, this potentially useful feature — which isn't mentioned in the larger Model 100 manual — doesn't seem to work.

LOOKING INSIDE

Although externally the same as the Model 100, the Tandy 102's a horse of a different color on the inside. For one thing, Tandy engineered the 102 with fewer parts. The circuit boards employ a newer surface-mount technology that, according to Mark Eppley, president of Traveling Software, "allows them to be stamped out like hot cakes off a griddle." That means that the Tandy 102's design is more efficient than the Model 100, implying that the machines are less expensive to manufacture. And, there are fewer parts to wear out.

The Tandy 102's system read-only memory (ROM) chip is soldered into the 102, as opposed to being snapped into a socket in the Model 100. This chip, which contains the operating system that drives the Intel 80C85 processor, tells the collection of chips, LCDs, transistors and relays how to act like a computer. With a non-removable system ROM, it appears that Tandy's software is now "etched in stone." Don't expect any future ROM upgrades here.

Just as in the 100, the 102 has three circuit boards connected by ribbon cables. One is for the keyboard, another for the LCD panel and the final one is the main or "motherboard."

What's instantly noticeable is that everything's more compact. The motherboard is angled, allowing the computer to be thinner. Its taller components are all located near the LCD side, and the smaller components appear toward the space-bar edge. It also has components on both sides, which helps save space.

The angled motherboard helps explain why items on the side on the computer (especially the bar code reader port and the DC power jack) look a little crooked.

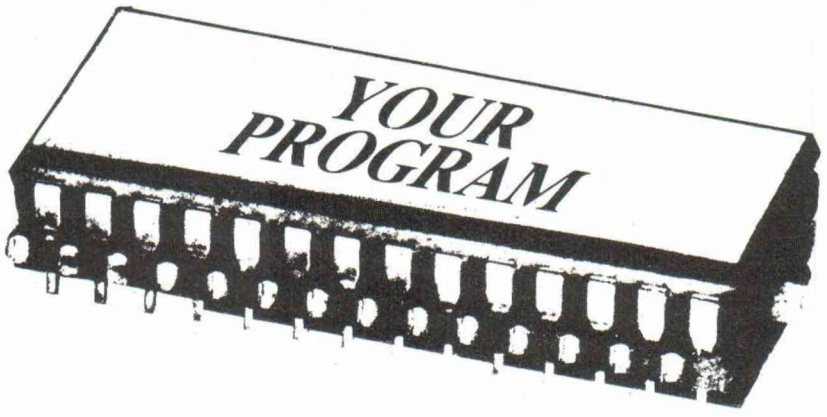
The circuit boards contain very few "cuts and jumps" that would indicate last-minute design changes. A handful of wires and capacitors have been added since the board was laid out, probably to reduce radio frequency (RF) emissions that would interfere with television and radios.

CARBON COPY

The system ROM in both the Model 100 and Tandy 102 is 32,768 bytes in size. Ninety-nine percent (or 32,413 bytes) are common between father and son.

There are a few software differences

Your own Program on a ROM



8K or 32K
Your BASIC or machine code program on a custom ROM ready to snap into your Model 100, Tandy 102 or Tandy 200.
You can't tell your program from a built-in. Starting at $50.00 each. Minimum order of 10. Quantity pricing as low as $15^00

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd. No. 207, Dallas, Tx. 75229

1-214-351-0564

VISA / MASTER CARD / C.O.D.

Circle 52 on Reader Service Card

between the old and new computers, however. For instance, in TELCOM on the Model 100, you can use the *find* function (F1) to extract a phone number for an ADRS.DO file. However, if you disconnect from TELCOM while a download is in progress, thereby returning to the TELCOM menu, subsequence attempts to use *find* result in an endless search of memory. In the 102, the *F1* key works every time.

There are few bugs that remain in place. The first is the mystery of the *ab* search. Create a document (.DO) file containing within it the string *aab*. Place the cursor at the beginning of the file, press *F1* and search for the string *ab*. You won't find it on either the Model 100 or Tandy 102.

Another bug is in Microsoft's BASIC, which still bears the Model 100 copyright. This bug occurs when executing a *GOTO* statement to a line number that's larger than 65529, but is still five digits long. No error message will result as long as there's a line in the program that matches the first *four* digits.

Type in and run the following program:

```
10 GOTO 88888
20 STOP
8888 BEEP
```

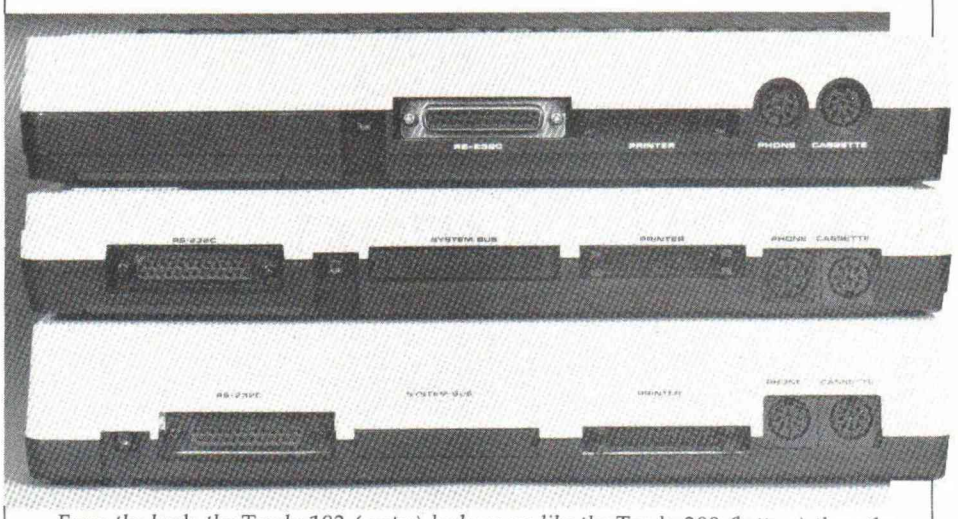
You shouldn't hear a beep. Instead, you should see an undefined line (UL) or overflow (OV) error since line 88888 is not only illegal but doesn't exist in the program. However, on both the 100 and the 102 the program jumps to line 8888.

And here's an amusing difference: Enter BASIC and type *DATE$ = "07/04/87"* Then press *F8* to return to the main menu. On the Model 100, the date will be *Jly 04,1987;* on the Tandy 102, you'll see *Jul 04,1987.*

Stuart Weinstock, Product Buyer for Tandy's portables, says there are some ROM improvements. These include an improved rounding scheme for certain BASIC calculations. Another is a control-character filter in TELCOM that protects against certain sequences that, in the 100, would disrupt the screen.

The input/output (I/O) ports are the same in the 102 as in the Model 100, but the hardware that enables them is different. Port decoding on the 100 required half a dozen chips. On the 102 it's collected into a special programmed-logic array that reduces chip count and thus assembly cost.

As in the 100, you can determine the position of the *DIR/ACP* switch in the 102 by testing bit 5 of input port 187. Similarly, in both machines you can also test the position of the *ORIG/ANS* switch through bit 4 of the same port. Finally, each computer receives bar code data through bit 3.



From the back, the Tandy 102 (center) looks more like the Tandy 200 (bottom) than the Model 100 (top).

SHOULD YOU UPGRADE?

Is the Tandy 102 an upgrade or merely a simple replacement for the Model 100? It's really more of a replacement. In other words, don't sell your Model 100 at a yard sale so that you can acquire the newer, sleeker model. And if you're shopping for a laptop, there's nothing wrong with picking up a used Model 100 at a good price. To be blunt, the Tandy 102 does nothing that the Model 100 doesn't do.

Compatibility between Model 100 and Tandy 102 hardware and software isn't an issue. ROM, disk and cassette-based software from all of the major vendors works without a hitch of both machines, according to representatives from Tandy Corp., Traveling Software and the Personal Computer Support Group (PCSG). Since the Tandy 102's option ROM socket is both physically and electrically identical to that in the Model 100, Model 100 ROMs such as Lucid, Ultimate ROM II, Write, Super ROM and Interactive Solutions work just fine in the 102.

Hardware is a different story. Nearly all Model 100 hardware works on the Tandy 102 — but some, such as PCSG/Holmes Engineering's Chipmunk disk drive, Tandy's Disk/Video Interface and Traveling Software's new Booster Pack require special cables to fit the Tandy 102's system bus connector. In those cases, the new design is an improvement.

In the RAM expansion arena, though, the new bus design is a drawback. Instead of burying RAM bank circuit boards, chips and batteries underneath the Model 100's option-ROM socket cover, these must plugged into the back of the laptop. This means that the RAM expansion must either hang off of the back of the computer — which is unsightly and insecure — or be located underneath the computer and be connected with a ribbon cable. If you're looking for esthetic multi-RAM bank expansions, you'll find them only on the Model 100.

What do industry sources think about the Tandy 102? "I take mine everywhere. The smaller size and lighter weight makes quite a difference," says Traveling Software's Mark Eppley. He adds, "Since it's 100 percent software compatible with the Model 100, the 102's perfect for going on the road."

Tandy Corp. obviously favors the Tandy 102, which has continued the Model 100's brisk sales. Few new laptop owners even realize there's a difference, according to one Radio Shack Computer Center manager. And of those who do, the fact that the world's best-selling and most useful laptop's even more convenient is a definite selling point in these days of high-powered MS-DOS laptops.

Here's the bottom line: Don't retire your Model 100 in favor of the Tandy 102. If you're shopping for a new laptop, though, the Tandy 102 is smaller, lighter, more affordable — and is an appropriate successor to the Model 100 dynasty. □

Housing a Fifth Nicad

I recently purchased a Model 100. However, here in Saudi Arabia alkaline batteries are expensive. So I thought I'd reconfigure the power supply to run off rechargeable batteries. But before I try it, I decided to consult an expert.

First I would put five 1.25 volt nickel-cadmium (nicad) batteries in a series by actually installing one of them inside the 100. I'd then require the AC adapter plug in the 100 so when I plug in the adapter, it would also recharge the batteries. Before I open the 100 up, please answer the following questions for me:

a. Is my idea realistic?

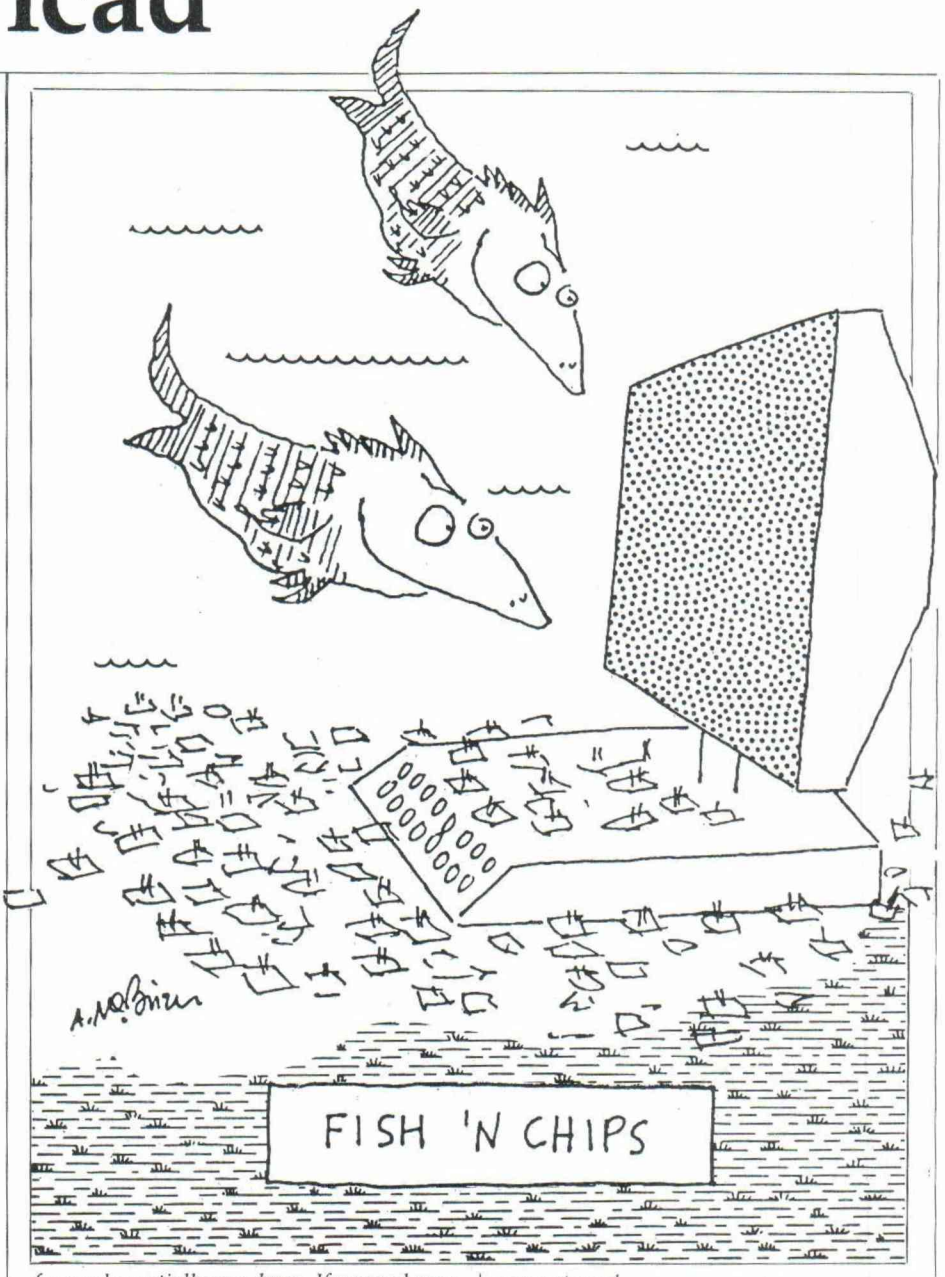
b. Can the AC adapter supply enough power to charge the batteries and run the computer at the same time?

c. Will there be any adverse affects to the power supply system or any other part of the computer during the charging mode?

d. Is there sufficient room inside the 100 to house the fifth nicad?

e. If there is sufficient room, should I shield, insulate or otherwise separate the battery from the rest of the computer?

Michael H. Ufnal
APO NY



We passed this one on to contributing editor Carl Oppedahl, author of "Inside the Model 100." His response:

You're not alone in wanting to use nicads in a Model 100 and let the AC adapter charge them up. As you are no doubt aware, four nicads trigger the 100's low battery light much sooner than four alkaline cells. The light goes on when the battery drops to about 4.1 volts. Four nicads, at 1.25 volts each, supply a total of only five volts. Therefore, a drop of just 0.9 volts will set off the light. Four 1.5-volt cells have to drop 1.9 volts to trigger the light. But five nicads, totalling 6.26 volts, will last quite a while.

The first thing to be careful of with nicads is that they be recharged and discharged in matched sets. Consider what happens with, say, one cell that's completely run down and four only partially run down. If you recharge all five in a series, the four recharged first can become overcharged. Doing this repeatedly will damage the nicads, and meanwhile the fifth cell may never get charged all the way.

Those four nicads suffer not only from overcharging but also from "discharge memory." If they get recharged before they are fully discharged, they lose the capacity that was not used.

Does this mean that you can't use five cells? Not at all. But it does mean that if you use a fifth cell hidden away in the computer, it and the four in the battery compartment must be inserted and replaced at the same time.

The AC adapter you use to power the Model 100, whether or not it's used for re-

charging, must contain a large (over 2000 microfarads) electrolytic capacitor for filtering. The Radio Shack adapter (Cat. No. 26-3804) contains such a capacitor. It's easily big enough to both power the computer and recharge the nicads.

You should be careful when placing the fifth nicad in the computer, since it might rattle around and damage other components. If you can solder wires on each end of the nicad (use a bit of sandpaper to rough up the metal first), this will save having to squeeze in a cell holder as well. A large flat cable runs between the LCD board and the main board. In my Model 100 it folds in such a way that it can surround and cushion a nicad cell located between connector CN7 and the existing cell holder. Otherwise you could put a carefully wrapped cell in the sparsely populated area around M32 and M35, just inside the printer connector (CN5). The cell need not be shielded magnetically but should be wrapped in plastic or electrical tape to prevent electrical contact with other parts of the computer. The cell should be in series with the four in the battery compartment.

For the AC adapter to recharge the nicads, you must provide a current path between pins 2 and 3 of the AC adapter connection. I would not use a dead short, but rather a small resistor, perhaps 100 ohms or so. There are ferrite beads and inductors on lines 1 and 2 of the adapter connector CN9 (covered with shrink wrap tubing), to keep the AC adapter's cord from becoming a broadcasting antenna. If you put the resister in the circuit at the printed circuit board, you allow RF interference to sneak past the beads and inductors and into the long cord of the AC adapter, which is bad news. It would be illegal to do this within the United States; I don't know if Saudi Arabia has RF emission limits.

If you put the resister right at the AC adapter connector, the interference will still be blocked. In any case, the ability of the adapter cord to act as an antenna is greatly cut down if you keep the cord bunched up rather than stretched out.

—Carl Oppedahl

THOSE JERKY INTERPRETERS

In BASIC programs with a lot of variables (or large arrays) program execution is jerky. While entering data the interpreter stops, goes off somewhere for a while and eventually comes back and finishes executing my keystrokes. Where is the "somewhere?" How can I minimize this annoyance? The only remedy I've been able to come up with is to buffer into a RAM storage file and keep the memory allocation for variables to a minimum.

Mark Campidonica
Davis, Calif.

If the program you're running is using the normal INPUT or LINE INPUT statement, your keyboard has the computer's undivided attention until you press Enter. Then the program processes the input and comes back to you for more.

A good programming technique is to have the computer ask most of its questions at the beginning of the program, instead of asking a question, doing a lot of processing, asking another question, doing more processing, and so on. Of course, not all programs lend themselves to that technique.

But if the program is using INKEY$ or the INPUT() function to read individual keystrokes, it might be checking each keystroke for validity before letting you continue. That might be the case if, for example, the program's prompting you for a telephone number — and it wants to reject all nonnumeric keystrokes. In that case, if the delay is too long, the program can be very difficult to use. The solution might be to modify the program's input routine to use the INPUT statement, instead of the key-by-key function, and test the validity of the data after everything's been typed in. —Ed.

LIMITED SPACE

I own a 24-kilobyte (K) Model 100, a portable disk drive and two printers. Presently, I'm interested in adding a ROM-based spreadsheet, either Multiplan or Lucid.

That brings me to a question about add-on chips. Since there is a limited amount of space (sockets) to add chips, will I be usurping space for future goodies? I'm confused about adding chips because I'm not sure about how much space they'll be using.

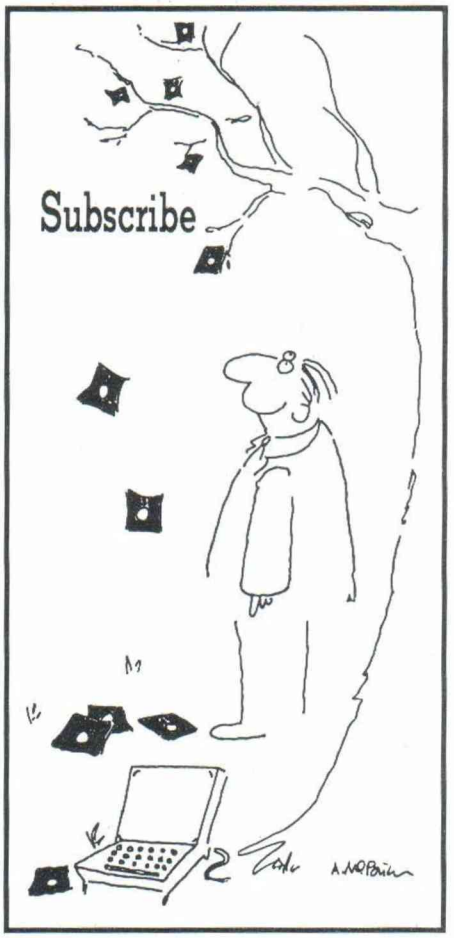
E.T. Dunlap
Oreland, Pa.

One of the best things about the option ROM modules like Multiplan, Lucid, Super ROM and Ultimate ROM II is that you can pop them in and pull them out as often as you please — the only concern being physical damage to the ROM's carrier.

If you buy a ROM today and decide that you want to change or upgrade a few months or years from now, just go through the old ROM's removal routine (as described in the owner's manual), pull it out and insert the new ROM.

If you want to have many ROMs available at one time, take a look at the Six ROM Bank from the Portable Computer Support Group.

But don't be concerned about using up room for future expansion. The option ROM socket can only be used for option ROMs (or adapters like the Six ROM Bank). —Ed.



A Software Bonanza

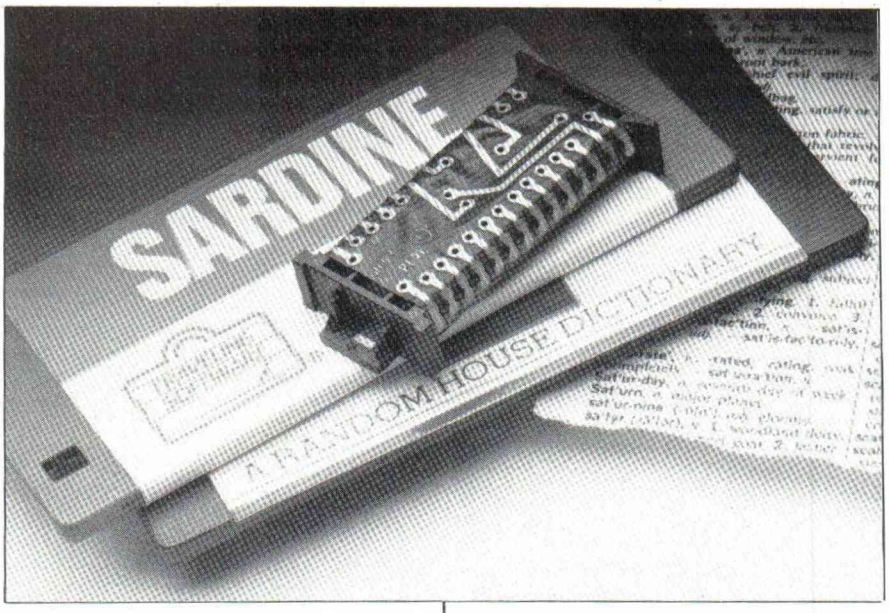
Topping the list of announcements in the last year from Traveling Software is Sardine, a laptop pocket dictionary and spelling checker. Available on 3.5-inch disk or on ROM, Sardine contains the entire Random House American Dictionary, over 33,000 words in all, and makes them accessible in seconds. According to Traveling Software's president Mark Eppley, "You can check a 25K document in about three minutes."

In addition, Sardine allows you to create your own auxiliary dictionary for adding any words, abbreviations or special terms you might use. The ROM version includes T-Word, Traveling Software's word processing program.

The disk version of Sardine sells for $99.95 and the ROM version is $169.95.

Next on the new product list is TS-Random, an upgraded version of TS-DOS, Traveling Software's operating system for the Tandy Portable Disk Drive.

TS-Random allows random access of up to eight BASIC files at one time. Additionally, it lets you format disks for different size format, allowing up to 1,600 records on a disk.



There are a couple of new housekeeping features too. One is a utility for recovering erased or damaged disk files. The other is a text compression program that lets you store as much as 150K on a 100K disk.

TS-Random comes on disk for $89.95 or on ROM for $119. TS-DOS owners can upgrade to TS-Random for $20 on disk or $49 on ROM.

For more information on Sardine or TS-Random contact Traveling Software, North Creek Corporate Center, 19310 North Creek Parkway, Bothell, WA 98011, (800) 343-8080.

Circle No. 80

Fix-It-All

While waiting for Tandy's hardware support personnel to return your call, you might try referring to a new book from Howard W. Sams and Co. called "Troubleshooting and Repair of Microprocessor-Based Equipment."

This 250-page manual provides a simplified, practical system of troubleshooting for many types of microprocessor-based electronic devices. The book includes numerous procedures and tricks that can be effective in diagnosing, isolating and locating faults in microprocessor circuits.

Topics covered include: microprocessor test equipment, stimulus response testing, register troubleshooting, current flow analysis, using the logic analyzer and more.

Written by John D. Lenks, "Troubleshooting and Repair for Microprocessor-Based Equipment" is available for $21.95 through bookstores, electronic distributors, or directly from Sams, 4300 W. 62nd St., Indianapolis, IN 46268, (800) 428-7267.

Circle No. 81

New Carrying Cases

The carrying case boom continues with several new models of silver-colored cases from Computer Coverup, Inc.

The cases are padded and are made

of lightweight Cordura nylon. They have velcro-sealed outer pockets for storing diskettes, tapes and other computer supplies. Each case includes an adjustable shoulder strap and reinforced handles.

The new silver cases are custom made to fit a variety of computers as well as most popular printers.

Computer Coverup produces a variety of products to protect, transport and improve the efficiency of microcomputers and printers in the office, home and in the field.

Prices of the new cases range from $65 to $90. Contact Computer Coverup, Inc., 2230 S. Calumet, Chicago, IL 60662, (800) 282-2541 or (312) 326-3000.

Circle No. 82

And On The ROM Front

Portable Computer Support Group has announced the release of Business Analyst, a new ROM based program for the Model 100 and Tandy 102.

Business Analyst performs detailed financial calculations such as break-even analysis and amortization at the touch of a function key. It also performs simple calculator functions.

Business Analyst ROM uses a feature known as Dynamic Tape to record all entries. This capability allows you to scroll an electronic "paper tape" on your laptop's screen.

Analyst Mode contains 10 registers for storing values such as present value, future value, interest rates, etc. It's capable of solving annuity and compound interest problems as well as computing net present value and return on investment.

The Business Analyst ROM cost $99.95 and is available from Portable Computer Support Group, 11035 Harry Hines Blvd., Dallas, TX 75229, (214) 351-0564.

Circle No. 83

Entertainment Online

Someone has finally taken the humdrum out of telecommunications and offered an online service that's both fun and informative. If you're into the stock market, being abreast of late breaking news or locating every instance of "heart transplant" in recent literature then this may not be for you. But then again....

ProtoCall is a new entertainment-oriented telecommunications service from Interplay, Inc. that, according to Dennis Flanders, Interplay's CEO, "extends the citizen bank radio setting by providing a more structured social backdrop." Among other things, that backdrop includes online interactive fantasy role-playing games.

"ProtoCall's interactive gaming area will provide an opportunity for people to create their own online personalities," explains Flanders, "and get acquainted in a variety of ways, whether they prefer international spying, slaying dragons or exploring uncharted galaxies."

ProtoCall is laid out as though it were a city, complete with streets, parks, beaches, a night club, movie theater, hotel and health spa. In addition to boating and sun bathing, subscribers share first-class fantasy flights from ProtoCall City's airport, making friends in real-time conversation, electronic mail and participating in their choice of special-interest groups (SIGs) and user-formed clubs. Private messages may be sent to anyone anywhere on the system and users can interact with automatons, artificially intelligent system-generated personalities that randomly roam about the city.

ProtoCall may be reached by telephone from over 600 U.S. cities without long-distance charges using any personal computer with terminal emulation software and modem. Connect-time fees for evenings, weekends and holidays are billed at $3.60 an hour at 300 bps, $4.80 an hour at 1200 bps. Weekdays from 7 a.m. to 6 p.m. are $12 and $15 per hour, respectively.

Contact Interplay, Inc., 10875 Main Street, No. 210, Fairfax, VA 22030, (800) 826-3286.

Circle No. 84

Rah Rah Sis Boom Bah

Soon it'll be time for football, fans and Model 100s. Model 100s? That's right. The Software Exchange has the Professional Football Handicapping System for football enthusiasts.

Available for both the Model 100 and Tandy 200, this cassette-based software lets you handicap weekly football games to help predict the winner. All information used is found in your daily newspaper's sports section. Enter the data and the program performs a statistical analysis to predict the winner, the point spread and the total points in the game.

The Professional Football Handicapping System is available for $39.95 plus $2 for shipping and handling. Contact the Software Exchange, P.O. Box 5382, West Bloomfield, MI 48033, (313) 626-7208.

Circle No. 85

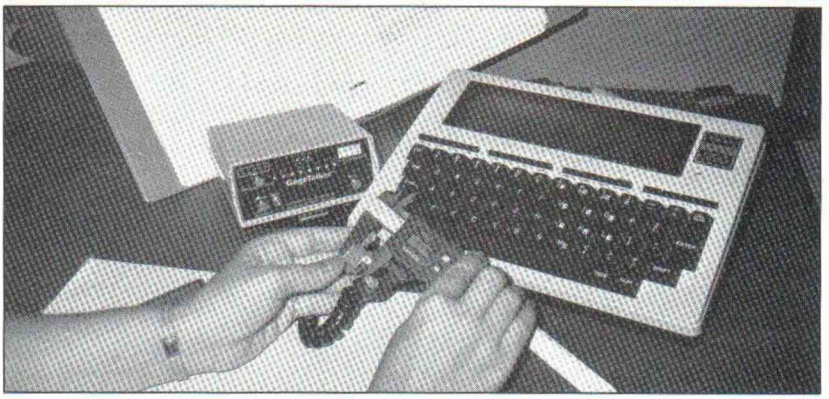
New Print Program

From Sun Belt Software comes EPRINT.BA; a word processor, formatter and typesetting program for the Model 100 and Tandy 200 computers. The program is designed for Epson FX printers, but the company is currently developing versions that will run on other printers.

EPRINT.BA allows use of nearly all the built-in Epson FX capabilities from control codes embedded in the document file. Any of several Epson print modes and international character sets can be selected and mixed as you write. Left, center, right and full justification, underlining, italics, double-strike, superscripts and subscripts, automatic centering, indentation, headers and footers, line spacing and margin spacing are but a few of the capabilities you can control.

Sun Belt's EPRINT.BA sells for $45. Contact Sun Belt Software, 2213 N. Main St., Las Cruces, NM 88001, (505) 514-3798.

Circle No. 86



Quality Control

For those of you interested in using your Model 100 or Tandy 102 in statistical process control to collect data from the manufacturing floor, analyze it and then transfer it to PC for further work and storage, a new program from the Crosby Company will be of interest to you.

The /SPC ROM Chip features the capability to receive direct input from a gauge using the RS-232 port, generate on-screen and printed average and range control charts and a printed process capability study.

The /SPC Chip is available as a stand-alone system or as part of the X&R Master Plus software package that works with the IBM PC and compatibles. Software is provided to transfer files to a desktop for SPC work. The package is available for $365, or $395 with X&R Master Plus included. Contact The Crosby Company, 564 Crescent Boulevard, Glen Ellyn, IL 60137, (312) 790-1711.

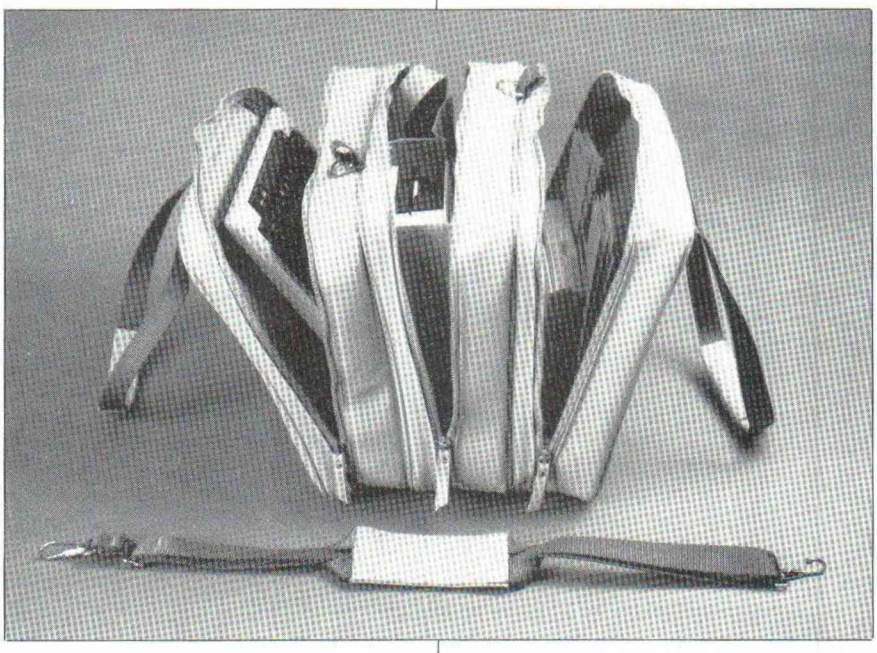
Circle No. 87

From ROMs to Cases

Portable Computer Support Group (PCSG), which brought you such products as Super ROM, Chipmunk Disk and the Six ROM Bank, now brings you something to carry them all in: a specially designed laptop computer carrying case.

The case measures 16 by 12 by 10 inches and is made out of the same "convertible top" canvas that's used on automobiles. There are three compartments designed to hold a laptop computer, portable printer, disk drive, assorted cables, papers and manuals.

Other features include an adjustable shoulder strap, leather handles, heavy-duty zippers and internal compartment pockets. It is padded inside and out with a closed-cell polyethylene foam.



The new case was designed by one of PCSG's corporate executives for the sole purpose of carrying his Model 100 and peripherals.

PCSG's Laptop Carrying Case is being sold for $69. Contact Portable Computer Support Group, 11035 Harry Hines Blvd., Dallas, TX 75229, (214)351-0564.

Circle No. 88

Talk To A Macintosh

Unlike a rose, a 3.5-inch disk isn't a 3.5-inch disk isn't a 3.5-inch disk. Data is formatted differently on the many varieties of microfloppy found today. An Apple Macintosh computer, for example, can't read a Tandy Portable Disk Drive (PDD) disk, despite the fact that the disks are *physically* identical.

So if you want to read Model 100 PDD disks on your Macintosh, or want to transfer a text file from the Mac to the laptop computer, forget sticking the PDD disk into the Mac. Instead, use a new product from Traveling Software: MacDOS.

MacDOS ($95) doesn't overcome the format incompatibilities between the Macintosh and the PDD. Instead, it allows you to plug the Portable Disk Drive directly into one of the Mac's serial communications ports and access the disk's files from a Macintosh window. From this window, which shows all PDD files and file sizes, you

can transfer, edit or print any or all files, just as you would any Macintosh data file. You have full file-editing capability for PDD files as large as 43K. MacDOS also offers an easy *Copy All* function that dumps all of the Portable Disk Drive's files to the Mac — great for using a Macintosh hard disk for laptop data storage.

MacDOS comes complete with special Macintosh-compatible adapter cable and software. Macintosh Plus and Macintosh SE owners will need a mini-8 to DB9 connector cable to use MacDOS.

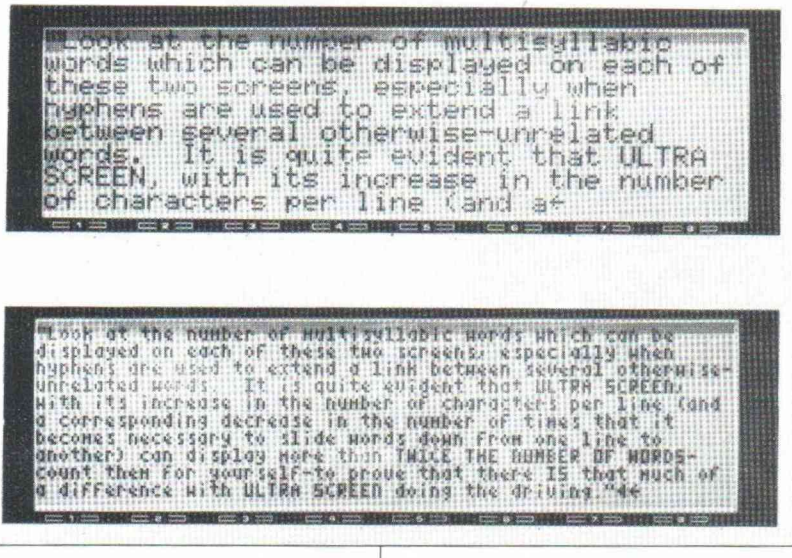
Contact Traveling Software, North Creek Corporate Center, 19310 North Creek Parkway, Bothell, WA 98011, (800) 343-8080 or (206) 483-8088.

Circle No. 89

Ultra Computing

UltraSoft Innovations has developed two new products: Ultrascreen, a screen enhancement program that almost doubles the amount of information displayed on a Model 100 or Tandy 102, and Disk Power II, a disk operating system (DOS) for the new 200K Tandy Portable Disk Drive-2 (PDD-2).

Ultrascreen (pictured) replaces the original 40 by 8 grid with a 60 by 10 display, while maintaining the standard character set, including graphics and international letters. It is available on either tape or diskette. Written entirely in machine language, it requires only 2.6 kilobytes of RAM. It can be used with the computer's built-in BASIC, Text, Telcom, etc., and can be relocated in memory for compatibility with other machine language programs.



Ultrascreen is priced at $29.95.

Disk Power II is an update on a previous Ultrasoft DOS. Just as the original Disk Power replaced Tandy's Floppy.CO for the 100K Tandy PDD-1, Disk Power II enhances the new PDD-2. Disk Power II supports the two-bank, 80-file configuration of the PDD-2 by providing two disk directories that can display all 40 files in each bank at once.

Disk Power II also makes it possible for 100/102/200 systems to download files that are larger in size than the amount of RAM available in the computer, storing them portion by portion to disk without disconnecting. This is designed to be of especial help for users connecting to online services.

Disk Power II is priced at $59.95.

For more information, contact Ultrasoft Innovations, Inc., 76 Main St., Champlain, NY 12919, (514) 457-9293.

Circle No. 90

Adding Menu Choices

When I started using the Model 100 after a couple of years of experience with my trusty Kaypro 4 desktop computer, the limit of 24K of memory quickly taught me lessons in being frugal with that precious resource called random-access memory (RAM). After having access to a megabyte of storage on floppies and on a RAM disk, working with only two-and-a-half percent of that amount really demanded efficient use and planning; eventually I got pretty good at it. As I became more proficient at using BASIC and writing compact utility programs plus a few fun ones, I ran smack into the other limit of the 100: directory spaces.

When it happened, there was 15K left in the machine, and no way to start my next writing project without killing a file or tagging it to the end of another text file. When I started looking at my BASIC files to determine which one to kill, I quickly came to the conclusion that I didn't want to live without any of them.

The solution was to combine the frequently used ones into one file. In addition to opening up directory spaces, this approach had the unexpected bonus of being able to use some common subroutines. The net result not only saved several directory spaces, but less memory was necessary for the programs as well.

It did take an evening of work — primarily changing BASIC line numbers to avoid conflicts between the programs — but the results were well worth the effort.

To start, list each program to the printer that you want to combine, then sit down with a colored pen and begin the notes on renumbering. The longest program will probably go first, and depending on the starting number, it may not have to be renumbered. The best way is simply to add two lines at the beginning of the program for each program that you combine. My longest program had line numbers from 10 to 360, so I reworked the first lines to have it begin at 40. I changed the next program to fall between 400 and 499. For the third program I used 500 through 599, and so on.

You can use the BASIC editor to change the line numbers, but if you have access to a program that does a search and replace, the process can be speeded up considerably.

Now comes the hard part. Go through the printed program listing again with another colored pen and find each and every GOSUB, GOTO, THEN or ELSE that refers to a line number. Replace each of these numbers with the new line number. Your chances of getting them all on the first go-round is slim, but the Model 100's BASIC will probably give you some clues when the program crashes.

After changing the line numbers, run each program through its paces and confirm that it is healthy. List the programs to the printer again to reconfirm that none of the programs share any line numbers.

Now it's time to build the skeleton of the program that will hold the whole mess. Name it something like UTILTY .BA. Here's what the beginning of my program looks like:

```
10 PRINT "1 - Calculator program"
11 PRINT "2 - Printer initialization"
12 PRINT "3 - RAM Bank overnight
   charge"
13 PRINT "4 - Dare"
14 PRINT "5 - WOW!"
15 PRINT "8 - Quit to Main Menu"
19 PRINT "SELECT ONE OF THE
   ABOVE"
20 GOSUB 9000
21 IF Y$ = "1" GOTO 40
22 IF Y$ = "2" GOTO 410
23 IF Y$ = "3" GOTO 510
24 IF Y$ = "4" GOTO 610
25 IF Y$ = "5" GOTO 710
26 IF Y$ = "8" THEN MENU
29 CLS
30 GOTO 10
```

Two notes: The GOTO statements should branch to the first line of the program, not to a remark. My line numbers for this menu are sequential, which is bad practice, but in this case it saved me from having to completely renumber the first, rather complex program. If you are experienced in BASIC, there are a number of ways to shorten and clean up the above. Here, it is listed in the simplest form for clarity.

Next, create the idler subroutine.

```
9000 Y$ = INKEY$
9010 IF Y$ = "" THEN 9000
9020 RETURN
```

This routine will wait for a key to be pressed, and when one is, it then returns with the value of that key to try to match with a GOTO between lines 21 and 26. If no match is found, lines 29 and 30 will effectively blink the screen and start the process over again. If you prefer letters to numbers, remember to allow for upper or lower case key entries. The line numbers for the idler subroutine are high to keep it at the bottom of the program. Then you can access it from anywhere in the combined programs.

Once you have completed this, save the skeleton file to UTILTY.BA.

To bring it all together, you must merge the programs that you started with, with the skeleton of the UTILTY program. To do this, they must first be converted to ASCII. This can more than triple the space taken by each file, so begin planning to unload some files on tape or other storage to make space if necessary. Type *NEW* and load a program to be included, then SAVE it to a .DO file by typing *SAVE "filename",A* and Enter. (Depending on the memory and directory spaces you have, you may have to go on to the next step, merge each file, then kill both the original program and the ASCII version before going to the next one.) Remember to back up your files on tape in case anything goes wrong.

Now the final step. Load "UTILTY." Type *MERGE "filename"* for each program to be included. When they are all loaded, run "UTILTY" through each of the included programs to confirm that they are all there and that they work properly. When you are satisfied, you can kill the original programs and the ASCII copies.

You can then go back and consolidate some of the subroutines and determine where the program goes after it's finished. For example, ending the program with MENU instead of END will dump the user back at the menu instead of presenting an *OK* prompt that can be confusing to persons intimidated by computers. To return to the UTILTY selection menu, end with a GOTO 10, or use a GOTO to send the user to one of the other programs you have included.

In my case, I ended up with four more directory entries, saved a few hundred bytes, and had a program that was friendly enough to be used by my non-technical roommate.

—Beverly Howard

A Disk Drive Shortcut

Despite some limitations, the Model 100's data-handling SCHEDL program and Tandy's affordable portable disk drive are wonderful tools. They're made even better when they work together, which they can now thanks to the program DB.CO.

Consider what you have to go through if you want to use your portable's built-in software to search a number of data files stored on disk. First, you access the disk drive's file manager, FLOPPY.CO. Next, you select File Load and enter the file name on disk, and then the file name you'll create in random access memory (RAM) — for this purpose NOTE.DO. When the file is loaded, you have to "Press Any Key" to get back to the file manager, and then press *F8* to get back to the Model 100 main menu. After that, access SCHEDL and search your data. When you need to look at a different data file on disk, you get to do that all over again. Every step. Every time.

Now consider the alternative offered here. When you run DB.CO, it immediately prompts you for the file name on disk. This file is immediately loaded to RAM as NOTE.DO with no other screens, prompts or keys needed. The program then jumps right into the SCHEDL program. Quite a difference.

LOADIT.BA, a BASIC program for loading DB.CO, a portable disk drive file management utility.

```
100 REM This program "pokes" DB.CO to RAM
110 CLEAR 256, 58999
120 X=59000
130 FOR J=1 TO 25
140 A=0
150 FOR K=1 TO 10
160 READ B
170 A=A+B
180 POKE X, B
190 X=X+1
200 NEXT K
210 READ C
220 IF C<>A THEN PRINT "DATA ERROR LINE"; (9000+(J-1)*10):END
230 NEXT J
240 PRINT "LOAD COMPLETE..."
250 END
9000 DATA  17,  81, 231,  33, 122, 232,  62,   3, 205,  98, 1084
9010 DATA  90,  33,  50, 232,  62,   3, 205,  98,  90,  33,  896
9020 DATA 105, 232,  62,   3, 205,  98,  90,  33, 203, 232, 1263
9030 DATA  62,   3, 205,  98,  90,  33, 119, 235,  62,   8,  915
9040 DATA 205,  98,  90,  33,  32, 239,  62,   6, 205,  98, 1068
9050 DATA  90,  33,  63, 239,  62,   6, 205,  98,  90,  62,  948
9060 DATA 101,  50, 105, 239,  42,  79, 231, 249, 195, 111, 1402
9070 DATA  91,  17,  71, 231,  33, 133, 246,  26, 119,  35, 1002
9080 DATA  19, 167, 194, 197, 230, 195,   5, 238,  57,  34, 1336
9090 DATA  79, 231,  62, 205,  50,  63, 239,  33, 191, 230, 1383
9100 DATA  34,  64, 239,  62, 195,  50, 203, 232,  50,  50, 1179
9110 DATA 232,  33, 122, 232,  34,  51, 232,  33, 120, 230, 1319
9120 DATA  34, 204, 232,  62,  55,  50, 122, 232,  62,  62, 1115
9130 DATA  50, 123, 232,  50, 124, 235,  62,  89,  50, 125, 1140
9140 DATA 235,  62,   1,  50, 124, 232,  62, 124,  50, 105, 1045
9150 DATA 239,  62,   0,  50,  32, 239,  50,  33, 239,  50,  994
9160 DATA  34, 239,  50,  35, 239,  50,  36, 239,  50,  37, 1009
9170 DATA 239,  50,  66, 239,  50,  67, 239,  50,  68, 239, 1307
9180 DATA  50, 126, 235,  50, 133, 246,  50, 105, 232,  50, 1277
9190 DATA 106, 232,  50, 107, 232,  50, 119, 235,  50, 120, 1301
9200 DATA 235,  50, 121, 235, 195,   8, 232,  78,  79,  84, 1317
9210 DATA  69,  46,  68,  79,   0,  43, 203, 205,  66, 114,  893
9220 DATA  33, 215, 232, 202,  77, 232, 205, 115, 236, 205, 1752
9230 DATA  68,  70, 209, 225,  58, 133, 246,  33, 205, 239, 1486
9240 DATA 205, 162,  17,  33, 192, 237, 205, 255, 237,   0, 1543
```

The program is written in machine language, but you don't need an assembler to use it. The BASIC program LOADIT.BA will load DB.CO for you. If you make a typing error in the DATA statements, the loader program will stop with an error message. For safety's sake, save a copy of the loader program to disk before you run it.

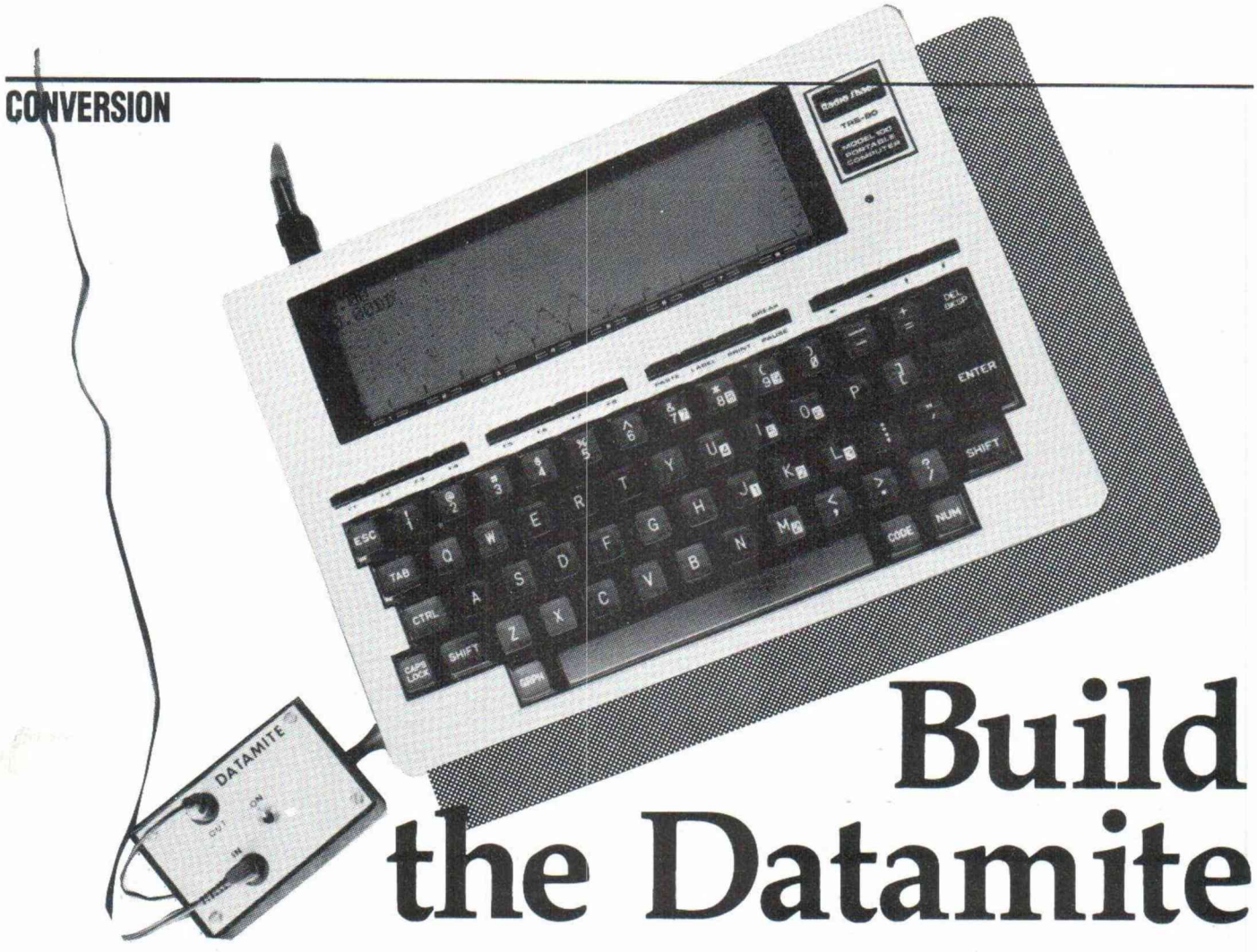
Once you've loaded the DB.CO program successfully, save it. Type in SAVEM "DB.CO",59000,59258,59088 to put DB.CO on your main menu. You should save it to disk too.

The program works by altering the disk drive's file manager — a form of brain surgery known as a "software patch." Then it jumps to the start of the file manager. With the patch in place, though, the manager behaves very differently.

For example, the file manager's menu isn't displayed. Instead of waiting for you to pick a function, it automatically selects File Load. After you enter the file name on disk, the altered file manager enters the name NOTE.DO as the file name in RAM, without prompting you. If there's already a NOTE.DO file in RAM, permission to overwrite with the new file is assumed.

The file now loads normally. When

Text continues on page 62



Build the Datamite

Low cost analog-to-digital conversion for the Model 100

By Kevin Jones

The small size and portability of the Model 100 computer make it uniquely useful for word processing, communication and computing. But imagine how the computer's usefulness would be enhanced if it could be used as a portable voltmeter, chart recorder, or frequency counter as well.

The Datamite is a simple gadget you can build to provide such additional versatility. The unit is small, 2.25 by 3.25 by 1.37 inches, operates for 80 to 100 hours on a single nine volt battery, and plugs into the Model 100's cassette jack. Best of all it can be built for less than $30.

I wanted to find a way to use the Model 100 to control the firing of my ceramics kiln so I could be free to do other work. I couldn't justify the expense of a fancy eight channel A/D converter since only one channel was needed, and the eight-bit resolution of many of the available devices was not adequate for the required temperature accuracy.

The solution to the problem of obtaining high resolution along with low cost presented itself in the form of a precision integrated circuit voltage controlled oscillator (VCO) manufactured by National Semiconductor, the LM-331. Used in conjunction with an operational amplifier and a few other components, the LM-331 converts varying voltage input to a square wave with frequency varying precisely in proportion to the input voltage. All that's needed in addition to the VCO to obtain a digital number that corresponds to this input voltage is a counter and a timer. The counter keeps track of pulses from the VCO for a known period of time and the resulting count is proportional to the voltage controlling the frequency of the VCO.

BUILT RIGHT IN

It so happens that the Model 100 cassette input is ideally suited as a frequency counter. Data from the cassette is transferred to and from memory in the form of pulses of varying length. The cassette input is connected to the serial data input of the 80C85 CPU in

Datamite Parts List

RESISTORS [1]

Part No.	Description
R1,8,10,11	10 kiloohm (K)
R2	1 megaohm
R3[2]	45.3K ¼ watt 1% metal film
R4[2]	11.3K ¼ watt 1% metal film
R5	2.2K
R6	100K miniature trim potentiometer (Mouser 32 RF 501)
R7	47K
R9[2]	8.87K ¼ watt 1% metal film
R12	2.2K
R13	5.6K

CAPACITORS

C1,6	1 microfarad (uf) 25 volt tantalum
C2	100 picofarad (pf) 50 volt disk ceramic
C3[3]	.0015 uf 5% 50 volt polystyrene film or monolithic (Mouser 23 PS 215)
C4[3]	.0068 uf 5% 50 volt polystyrene film or monolithic (Mouser 23 PS 268)
C5	.22 uf 50 volt monolithic (Mouser 581-UDW224M1)

ICs, TRANSISTOR

IC-1	CA 3130 E (RCA) (Mouser 570-CA3130E)
IC-2	LM 331 N (National Semiconductor)
Q1	2N3906 (Mouser 333 KN 3906)

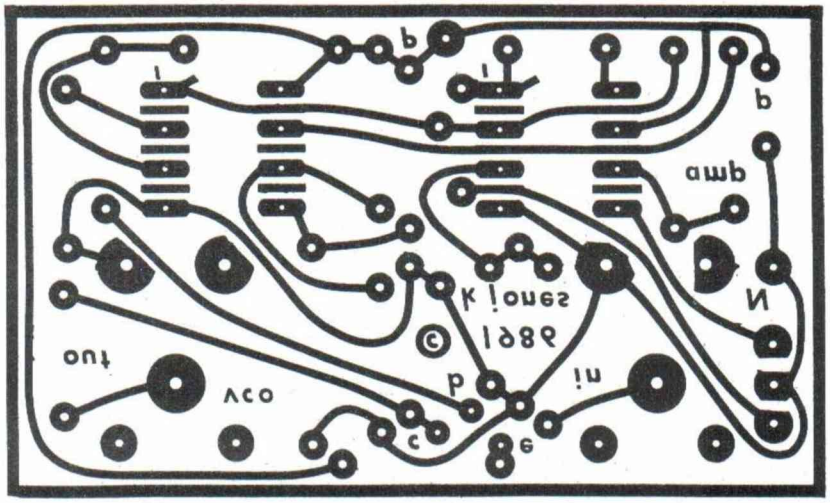
MISCELLANEOUS

P1,2 (Mouser 12 BC 124)	snap-type 9 volt battery connector
J1,2	miniature phone jack (Mouser 16 PJ 528)
J3	optional sub-miniature phone jack (Radio Shack #274-292)
S1	SPST miniature on-off toggle switch (Mouser 10 TC 220)
B1	9 volt alkaline battery
Cabinet	2.20 by 3.35 by 1.12-inch plastic case with panel (Mouser 546-1591S-BK)

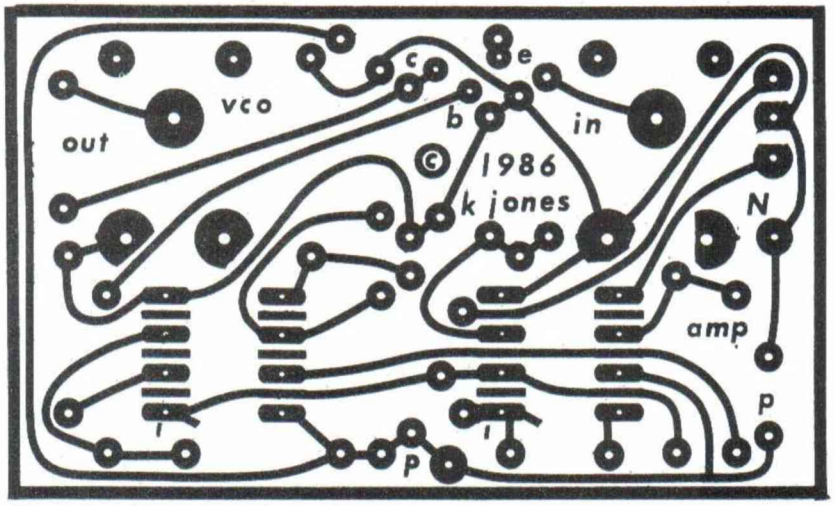
[1] Unless otherwise noted, all resistors are ¼ watt 5% carbon film.
[2] These resistors may vary 5% from stated values, but must be 1% metal film types for temperature stability.
[3] These capacitors must be temperature stable types.

Parts bearing Mouser numbers obtainable from Mouser Electronics, 11433 Woodside Ave., Santee, CA 92071, (619) 449-2222.

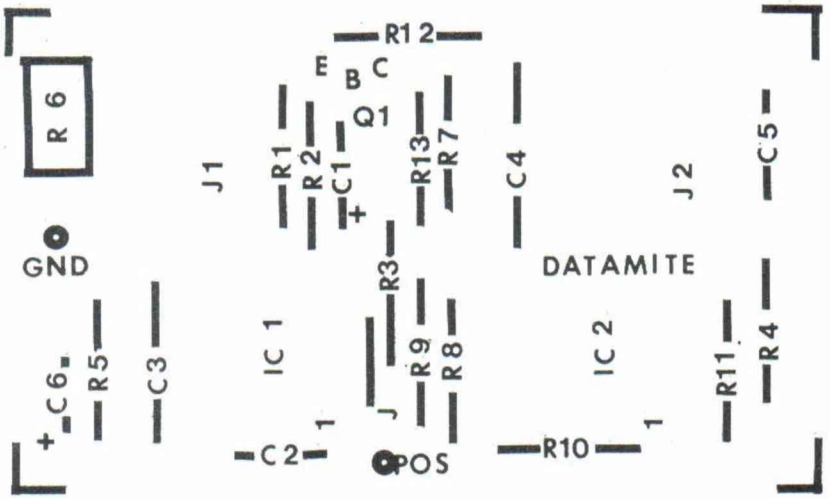
Author's Note: A parts kit including all components listed above (except battery & J3), with circuit board and panel pre-drilled is available from Jones Service & Design, 1842 S. Nugent Rd., Lummi Island, WA 98262, (206) 758-7258. Price of the parts kit alone is $29.50; kit with voltmeter software on cassette is $34.50; Datamite assembled with voltmeter and chart-recorder software on cassette is $62.50. Please include $3 for postage and handling.



Circuit board layout seen from component side.



Circuit board layout seen front side.



Component location overlay for top of board.

the Model 100, which can be programmed to detect the presence or absence of an input pulse. A short software routine enables the CPU to count these pulses for a controlled interval and deposit the result in two bytes of memory that can be accessed from BASIC.

An eight-bit digital word provides a *resolution* of 256 counts. In other words, if an eight-bit register in the Model 100 is used to count to 255 in a measured time period, and that count corresponds to a 5.1 volt input on the VCO, then the voltage resolution is 5.1 volts divided by 255, which is 0.02 volts per count.

For some applications such as a chart recorder, this is entirely adequate, but for temperature measurement in a kiln which must be accurate to within two degrees in 2,000 degrees, more resolution is required.

The disadvantage of high resolution is that it requires a longer period to gather a large number of counts. For example, if the VCO is operating at a frequency of 10KHz, which is near the maximum rate that can be processed by the Model 100 cassette input, the sample time necessary to fill sixteen bits is 6.4 seconds. A more reasonable sample interval is one second, corresponding to 13 bits resolution.

The frequency-counter software used here is flexible in that the count period may be adjusted over a wide range so that a relatively fast sample may be obtained with only eight bits resolution, or a longer sample may be taken resulting in higher resolution. The circuit to follow provides a voltage-frequency conversion accuracy to within a few counts in 10,000. Since the counting period is determined by the crystal-controlled clock in the Model 100, an accuracy of up to one millivolt in five volts may be expected.

The Datamite may be built from the parts kit provided by Jones Service and Design (see parts list) or hand-wired on a perforated circuit board. If you're familiar with printed circuit fabricaton, you may make your own board from the layout shown here. The following instructions assume you are using the kit.

Assemble the printed circuit board by pushing the leads of the components through the labeled holes in the board, referring to the parts list for values. As each component is inserted, press it snugly to the top of the board with one finger and bend the leads outward where they protrude from the

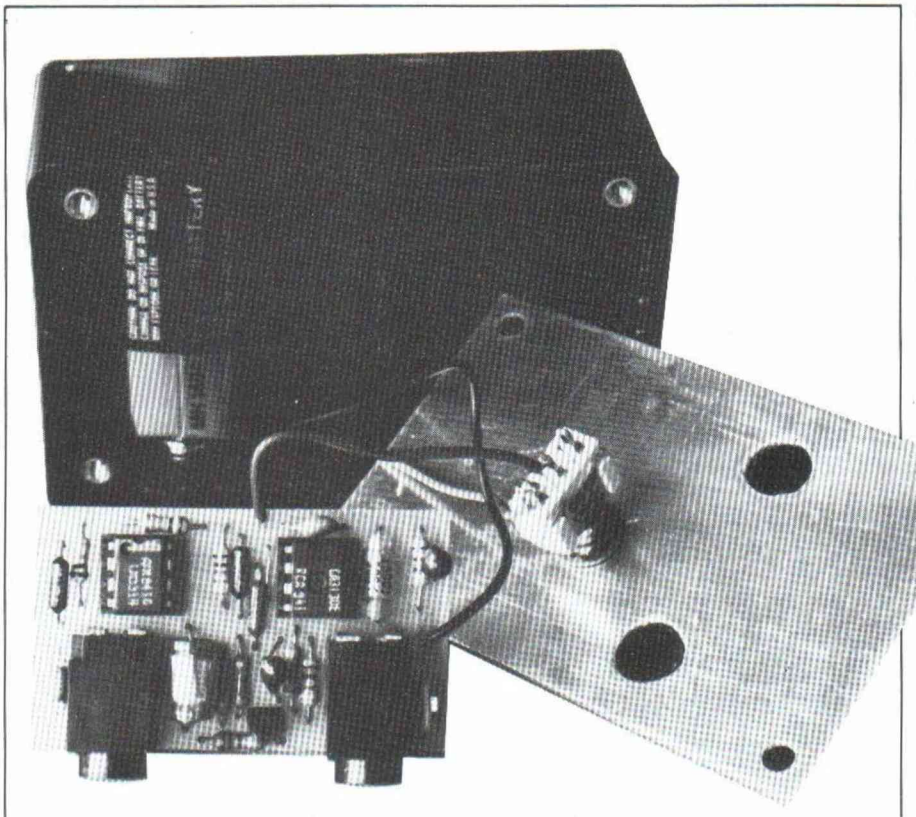
bottom to hold the component in place. Observe the polarity marked on the board for C1 and C6, and be sure the IC's are inserted with pin one in the correct position. The offset null potentiometer, R6, is mounted with the slider toward the inside of the board. The two holes labeled J between R3 and IC-1 are connected with a wire jumper. When all components have been inserted, turn the board over and solder the connections using a low-power iron with a one-sixteenth-inch tip, being careful not to make solder bridges between adjacent connections. Cut off the protruding leads. Solder the black wire from the battery connector to the hole in the board marked GND and a two-inch length of red wire to the hole marked POS.

After the circuit board wiring is completed, lay it aside temporarily. Remove the protective covering from the cabinet panel, peel the paper backing from the panel decal, and stick it in place. With a sharp knife cut the decal material out of the holes for the switch and jacks. Mount the switch and solder the two red wires from the battery to the two switch terminals closest to the bottom of the panel.

The optional external power jack, J3 on the schematic, may be mounted in a hole drilled under the switch. Since the battery life is quite long, this jack is only necessary if you anticipate leaving the Datamite on for many hours at a time. Slide the circuit board into the case so that the components face the switch and the jacks will line up with the holes in the panel. The battery fits vertically on the right, its bottom sliding into the space where the ridges have been cut from the bottom of the case.

The LM-331, IC-2, contains a switched precision current source, a comparator and a one-shot timer. The CA-3130A, IC-1, is connected as an integrator that charges C3 at a rate dependent on the voltage applied to pin three, its non-inverting input. The integrator output, pin six of IC-1, is fed to the positive comparator input of IC-2 via pin seven.

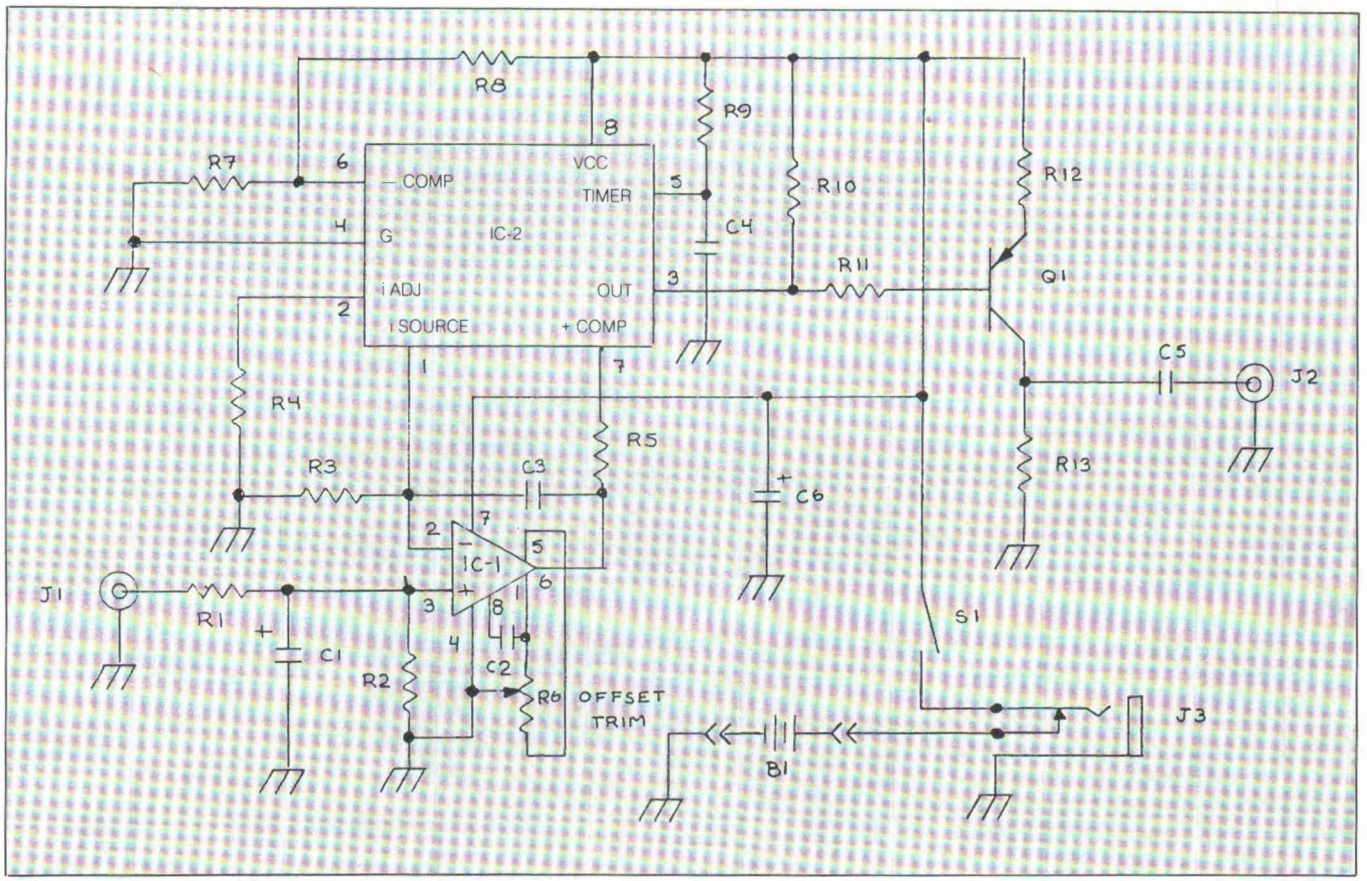
When this output reaches a value equal to the voltage supplied by R7/R8 to the negative comparator input, pin six of IC-2, the one-shot timer is triggered. The timer stays on for a period determined by the values of C4 and R9. While it's on, the output of IC-2, pin three, is held low and the switchable current source is turned on. This current flows out of pin two, reversing the charging of C3, thereby causing the



The inner workings of the Datamite.

```
10 ' "VLTMTR.BA", including machine language portion in
20 '  DATA statements copyright 1986 Kevin G. Jones
30   CLS:PRINT:PRINT" A/D.BA LOADS A MACHINE LANGUAGE"
40   PRINT" PROGRAM INTO MEMORY FROM 62900-62958"
50   PRINT:PRINT" DO YOU WISH TO SAVE ANY DATA IN":PRINT
" THIS AREA?   Y/N";
60   SV$=INKEY$
70   IF SV$="Y" OR SV$="y" THEN CLS:PRINT:PRINT" SAVE
DATA AND RESTART A/D.BA":STOP
80  IF SV$="N" OR SV$="n" THEN 100 ELSE 60
90  IF HIMEM>62899 THEN CLEAR 256,62899
100
CK=62900:AL=CK+55:AH=CK+56:CL=CK+57:CH=CK+58:KV=.5E-3
110  FOR DR=CK TO (CK+54):READ DT
120  IF DT<>PEEK(DR)THEN 140
130  NEXT DR: GOTO 170
140  RESTORE
150  FOR DR=CK TO (CK+58):READ DT:POKE DR,DT
160  NEXT DR
170  CLS:PRINT:PRINT
180  PRINT" 1. MEASURE VOLTAGE OR CALIBRATE":PRINT" 2.
FREQUENCY
190  PRINT" 3. USER PROGRAM":PRINT" 4. QUIT
200  SV$=INKEY$
210  IF SV$="1" THEN GOSUB 510
220  IF SV$="2" THEN GOSUB 620
230  IF SV$="3" THEN GOSUB 750
240  IF SV$="4" THEN CLS:STOP
250  IF SV$="" GOTO 200 ELSE GOTO 170
```

Listing continues on page 25



Schematic of the Datamite VCO unit.

output of the integrator, pin six of IC-1, to ramp downward. When the timer finishes its cycles, the output of IC-2, pin three, reverts to its high state, the current source turns off, and the cycle repeats with the integrator output once more ramping upwards. Q1 inverts the output of IC-2 and provides a voltage level compatible with the cassette input of the Model 100.

Type in the BASIC program listing VLTMTR.BA shown here, or load from cassette if you have purchased it with the parts kit. If you type in the program manually, double check the accuracy of the data statements in lines 710-730. These constitute the machine-language frequency counter portion of the software that is poked into memory from 62900 to 62958. If you make a mistake entering the data statements you could get a cold start, so save your files as insurance.

Plug the black cable from the cassette cable into the Datamite output. Run VLTMTR.BA and connect an accurately known calibration voltage over one volt to the input. A good calibration source is a fresh 1.35 volt mercury cell or you may use a regulated power supply and an accurately calibrated DVM. Make sure the polarity of the input voltage is positive because the VCO will not respond to a negative voltage.

The Model 100 cassette input is ideally suited as a frequency counter.

Select "Measure Voltage or Calibrate" by hitting *1* at the VLTMTR.BA menu. A voltage will be printed on the screen. Disconnect the voltage standard from the input of the Datamite. The voltage indicator on the screen should drop to near zero. Adjust the offset adjustment, R6 on the Datamite, until you get a reading of a millivolt or two, then slowly back off until the reading is *just* zero, no farther. Reconnect the calibration source. Press C. The program will ask if you want to calibrate, then will ask for the voltage. For a fresh mercury cell, type in 1.351 then Enter. You will be returned to the voltage display that will now read the correct voltage.

For frequency measurements, disconnect the black cassette plug from the Datamite and connect it to the unknown frequency source (a symmetrical square wave under 10 kilohertz and of about one volt amplitude works best) through a .22 microfarad capacitor. Select "Frequency," number 2 on the menu, and the frequency will be displayed. The calibration here depends on the frequency of the CPU clock in the Model 100, and should be accurate to within a few hertz.

If you wish to fine-tune the calibration and have an accurately known frequency source, you may do so by adjusting the sample time that is controlled by the BASIC variable RS in line 630 of the program. Increasing or decreasing this number will proportionately increase or decrease the sample time, and therefore the frequency calibration. If you wish to preserve your voltage calibration, return to the menu

```
260 '
270 'CALIBRATE
280 '
290 FK$=INPUT$(1)
300 CLS:PRINT:PRINT"  DO YOU WISH TO CALIBRATE? Y/N";
310 FK$=INPUT$(1)
320 IF FK$="N" OR FK$="n" THEN 440
330 IF FK$="Y" OR FK$="y" THEN 340 ELSE GOTO 300
340 CLS:PRINT:INPUT"  KNOWN VOLTAGE AT INPUT";VR
350 GOSUB 460
360 IF VN<1 THEN 370 ELSE 400
370 CLS:BEEP:PRINT:PRINT "  REFERENCE VOLTAGE UNDER 1
VOLT"
380 FOR N=1TO 500:NEXT N
390 GOTO 440
400 SC=VR/VN
410 SN=(PEEK(CL))+(256*PEEK(CH)):RS=INT(SN*SC)
420 R2=INT(RS/256):R1=RS-(256*R2)
430 POKE CH,R2:POKE CL,R1
440 RETURN
450 '
460 CALL CK:CT=PEEK(AL)+256*PEEK(AH)
470 VN=CT*KV:RETURN 'CHECK INPUT,COMPUTE V
480 '
490 ' MEASURE VOLTAGE
500 '
510 CLS
520 GOSUB 460
530 SV$=INKEY$
540 IF SV$="C" OR SV$="c" THEN GOSUB 300:GOTO 510
550 IF SV$<>"" THEN RETURN
560 PRINT:PRINT:PRINT"      V=";:PRINT USING "#.###";VN
570 PRINT:PRINT"          ANY KEY FOR MENU":PRINT "
C=CALIBRATE"
580 CALL 23914:GOTO 520 'HOME CURSOR
590 '
600 ' FREQUENCY
610 '
620 CLS:SH=PEEK(CH):SL=PEEK(CL)
630 RS=23644
640 GOSUB 420
650 GOSUB 460
660 SV$=INKEY$:IF SV$<>"" THEN 700
670 PRINT:PRINT:PRINT"    FREQ=";:PRINT USING "#####";CT
680 PRINT:PRINT"       ANY KEY FOR MENU"
690 CALL 23914:GOTO 650
700 POKE CH,SH:POKE CL,SL:RETURN
710 DATA
243,1,0,0,42,237,245,235,195,194,245,62,0,3,27,122,179,2
02,227,245,32
720 DATA
7,218,209,245,35,195,194,245,62,0,3,27,122,179,202,227,2
45,32,7,210,191
730 DATA
245,35,195,212,245,33,235,245,113,35,112,251,201,0,0,89,
92
740 '
750 RETURN 'USER PROGRAM
```

from frequency mode before interrupting the program. This is because voltage and frequency modes poke different sample times into the machine-language frequency counter program and the voltage sample time is replaced when the program leaves the frequency subroutine and returns to the menu.

A user space is provided at line 750 in the software where you may insert your own analog-to-digital conversion programs. If you want your program to measure the input voltage of the Datamite, put in the line GOSUB 460. The BASIC variable VN will then contain the present value of the input voltage.

You can control external devices using the motor relay in the Model 100 (the smaller gray plug at the end of the cassette recorder cable). Don't try to control any current greater than 500 milliamps, or the relay could be damaged. For example, suppose you wish to use the motor relay to light an LED if the input voltage to the Datamite is greater than two volts. You would use the following program:

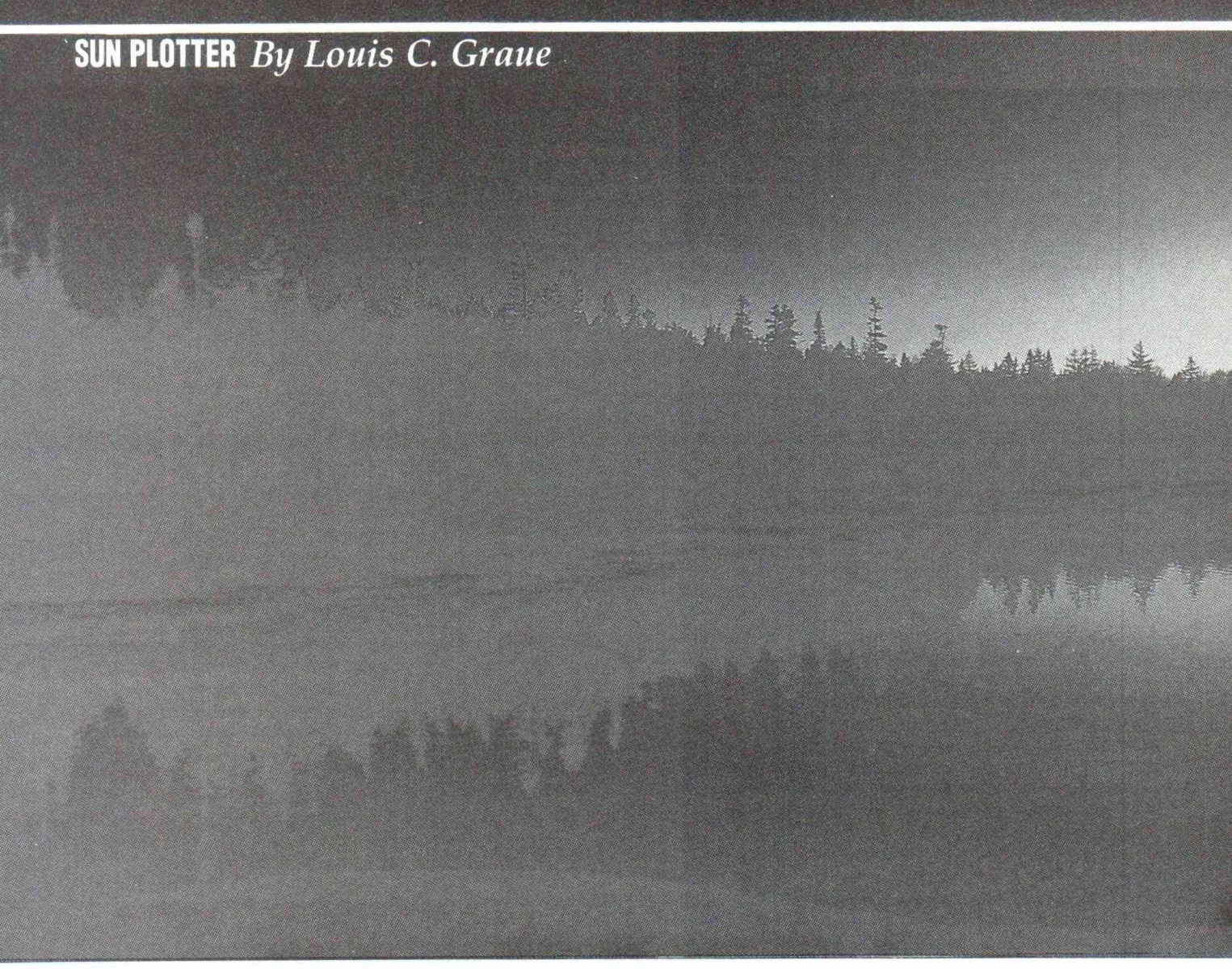
```
750 GOSUB 460 : IF VN>2 THEN
    MOTOR ON : GOTO 750 ELSE
    MOTOR OFF : GOTO 750
```

If you select User, program number 3 on the menu, the above routine will turn the motor relay on any time the input voltage is more than two volts.

The Datamite has endless applications. By the addition of voltage dividers or a simple operational amplifier, the input voltage range can be extended from millivolts to hundreds of volts, or it may be made to measure any other parameter that can be converted to a voltage: temperature, wind speed, barometric pressure, rainfall, illumination and skin resistance, to name a few. Besides employing the Datamite to fire pottery, I have used it with a simple four-channel multiplexer to record weather data, and with the chart-recorder program mentioned in the parts list to graph micro-variations in barometric pressure. A project I'm contemplating for the future is a "Model 100 enlarging computer" for the photographic darkroom.

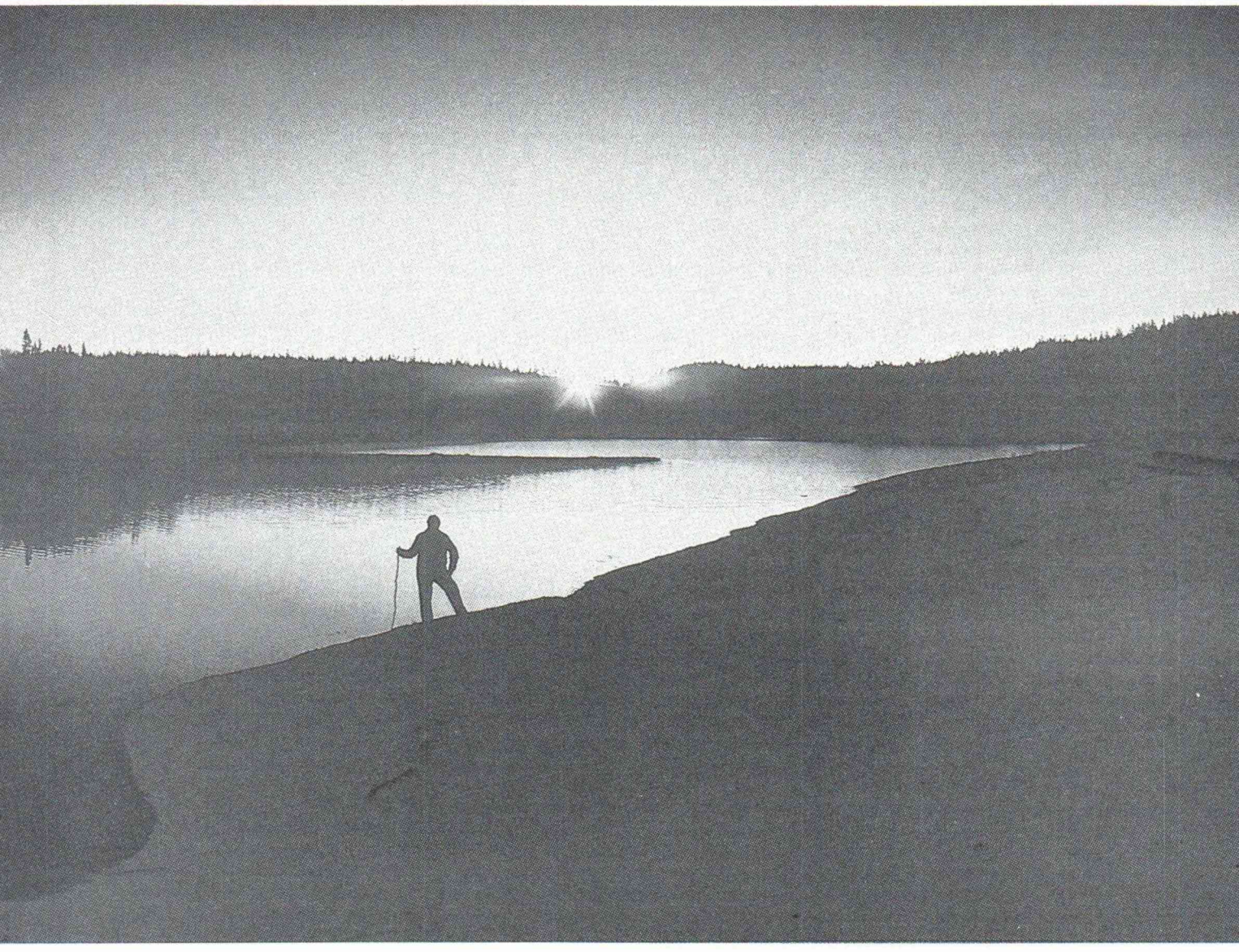
So far, the only disadvantage found to owning the A/D is that the Model 100 never seems to be available for letter writing and account-keeping because it is tied up firing pottery or taking data. □

SUN PLOTTER *By Louis C. Graue*



Here Comes the Sun

Use your portable to track Sol's journey across the daytime sky.



This program for the Model 100 calculates the azimuth and altitude of the sun for every hour of the day at a specified spot on the earth. The date, latitude, longitude, times, azimuths and altitudes are printed right on the screen. In addition, the path of the sun during the daytime is plotted against local time. The times of the local sunrise and sunset for the date are also printed on the screen.

The program is mostly just for fun, but has some practical uses as well. If you wanted to plan window shading then the solar coordinates for various times of the year might be very useful. Modifications could be made to track the sun's path for a solar energy device. Fisherman and hunters can determine the legal times to operate, accident reports involving sun glare can be documented, and so on.

You can estimate the local time of the maximum altitude of the sun. When the sun crosses your meridian of longitude, it is due south of you. Hence, the shadow cast by a vertical object will point true north. This can then be used to calibrate your rotator for your television or radio antenna.

The method for finding solar coordinates is given in the *Almanac for Computers 1980,* published by the Nautical Almanac Office.

The calculations use the current months, day and year to generate the Julian date, Greenwich mean time, sidereal time, Greenwich hour angle and local hour angle. Trigonometry is then used to produce the solar azimuth and altitude.

Greenwich is a borough of London, England. Mean solar time for the meridian at Greenwich is used as a basis for calculating time throughout most of the world. Julius Caesar introduced the Julian calendar in Rome in 46 B.C. The Julian date is a count of days and fractions of days beginning at Greenwich Mean Noon, January 1, 4713 B.C. The formula for calculating this from the current date is given in line 390 of the program.

Sidereal time is based on the axial and orbital rotation of the earth with reference to the background of stars. A sidereal day is 23 hours, 56 minutes, 4.09 second in units of mean solar time.

Formulas from the Nautical Almanac are used to calculate the right ascension and declination of the sun. These are then used to with the local longitude and latitude to calculate the azimuth and altitude of the sun at that location. To obtain a good understanding of how this is done requires some study of an astronomy or navigational textbook.

Lines 100 through 195 set up the screen with a coordinate system and headings. After the calculations are made, data is displayed and points are plotted for each hour of the day within this framework. The constant KC = 57.29578 in line 310 is used to convert degrees to radians.

PHOTOGRAPH BY BENJAMIN MAGRO

Lines 350 through 370 pick out the month, day and year stored in DATE$, which is maintained by the Model 100. Line 380 converts the year from two to four digits and line 390 calculates the Julian date.

The FOR-NEXT loop from line 400 through line 670 calculates the sun's azimuth and altitude for each hour of the day. The variable UT is Universal (Greenwich) time and ES is the corresponding Eastern Standard Time. I will explain them to you so you will be able to change to your own local time. Zero hours UT is 1900 hours EST. Five hours UT is 0 hours EST; EST is five hours behind UT. In the program we run UT from 5 to 29 so that EST will run from zero to 24 hours. This is necessary so that the graph will be correct. For example, to adjust from the Pacific Standard Time (PST) zone we use the fact that PST is eight hours behind UT. Lines 400 and 410 should be changed to read as follows:

```
400 FOR UT = 8 TO 32
410 ES = 1600 + UT * 100
```

If you want the output to agree with your watch and hour watch is "h" hours less than UT, then subtract "h" hours from ES and run UT from "h" to h+24.

Line 658 sends us to a subroutine that updates the display of local time, altitude and azimuth numbers. We come back to line 659 where we check for a change in the sign of the sun's altitude. If there is a change we go to a subroutine to calculate the time of sunrise or sunset. If this is the first change then the time of sunrise is computed. If this is the second change then the time of sunset is computed. This subroutine uses linear interpolation between the two altitudes to get the approximate time of zero altitude. Since the sun's path is concave downward, these approximations will give a sunrise time slightly later and a sunset time a little earlier than true values.

If you want to obtain very precise sunrise and sunset times then you can write a program using the main calculating loop repeatedly to search for the time when the sun's altitude is zero.

Line 665 checks for a positive altitude. If it is found to be positive, then the sun is above the horizon and we go to a subroutine that plots the sun's altitude versus local time.

```
100 CLS
110 PRINT"     SOLAR ALTITUDE"
115 PRINT"90";
    :PRINT@200," 0";
116 PRINT@242,"0";
    :PRINT@246,"6";
    :PRINT@250,"12";
    :PRINT@254,"18";
    :PRINT@258,"24";
120 LINE(14,8)-(110,44),1,B
130 FORI=12TO40STEP4
    :PSET(15,I)
    :NEXT
135 FORI=12TO40STEP4
    :PSET(109,I)
    :NEXT
140 FORI=18TO106STEP4
    :PSET(I,43)
    :NEXT
150 FORI=38TO86STEP24
    :PSET(I,42)
    :NEXT
160 PRINT@22,"DATE ";DATE$;
170 PRINT@60,"";
175 LA=41.35
    :LO=83.63 '(replace with your coordinates)
180 PRINTUSING"Lat=##.## Long=##.##";LA,LO
185 PRINT@100,"";
190 PRINT"Time Azmuth Altitude";
192 PRINT@140,"";
193 PRINT"---- ------ --------";
195 PRINT@180," +++ COMPUTING +++";
285 I1=6
310 KC=57.29578
    :SL=SIN(LA/KC)
    :CL=COS(LA/KC)
320 '
330 ' *** CALCULATE JULIAN DATE
340 '
350 M=VAL(LEFT$(DATE$,2))
360 I=VAL(MID$(DATE$,4,2))
370 K=VAL(RIGHT$(DATE$,2))
380 K=K+1900
390 JD=367*K-INT(7*(K+INT((M+9)/12))/4)+INT(275*M/9)
+I+1721013.5-.5*SGN(100*K+M-190002.5)+.5
400 FOR UT=5 TO 29 '(adjust for timezone)
410 ES=1900+UT*100 '(adjust for timezone)
420 IF ES>=2400 THEN ES=ES-2400
430 JT=JD+UT/24
440 T=(JT-2415020)/36525
450 MS=358.476+35999.05*T
460 L=279.691+36000.769*T
470 LS=L+(1.919-.0048*T)*SIN(MS/KC)+.02*SIN(2*MS/KC)
480 IF LS>360 THEN LS=LS-360
490 IF LS>360 THEN 480
500 EP=23.452-.013*T
510 TR=COS(EP/KC)*TAN(LS/KC)
520 SD=SIN(EP/KC)*SIN(LS/KC)
530 RA=ATN(TR)*KC
540 IF RA<0 THEN RA=RA+180
550 IF ABS(LS-RA)>90 THEN RA=RA+180
560 IF RA>360 THEN RA=RA-360
570 DE=ATN(SD/SQR(-1*SD*SD+1))
    :CD=COS(DE)
580 GM=6.67170278+.0657098232*(JD-2433282.5)+
1.0027379093*UT
590 OM=372.1133-.0529539*(JT-2433282.5)
600 E=.00029*SIN(OM/KC)
610 GA=GM+E
620 LH=15*(GA-RA/15)-LO
    :SH=SIN(LH/KC)
    :CH=COS(LH/KC)
630 SA=SL*SD+CL*CD*CH
640 AL=ATN(SA/SQR(-1*SA*SA+1))*KC
```

```
655 AZ=ATN(SIN(LH/KC)/(COS(LH/KC)*SIN(LA/KC)-TAN(DE)
*COS(LA/KC)))*KC
656 IF AZ<0 THEN AZ=AZ+180
657 IF ES>1200 THEN AZ=AZ+180
658 GOSUB 900
659 IFA1*AL<0THENGOSUB 950
    :S1=1
660 A1=AL
665 IF AL>0 THEN GOSUB 690
670 NEXT UT
680 GOTO 780
690 '
720 I1=I1+1
730 T1=ES/100
740 C=4*T1+14
750 R=INT(44-.4*AL+.5)
760 PSET(C,R)
770 RETURN
780 PRINT@224,"";
    :PRINTUSING"Sunrise ####";SR
790 PRINT@264,"";
    :PRINTUSING"Sunset  ####";SS
800 GOTO 800
900 PRINT@180,"";
905 IF ES<100 THEN 920
910 PRINT USING"####  ###.#    ###.#";ES,AZ,AL
920 RETURN
950 MO = INT(60*ABS(A1)/(ABS(A1)+AL)+.5)
    :EV=INT(60*A1/(ABS(AL)+A1)+.5)
960 IFS1=0THENSR=ES-100+MO
    :IFMO=60THENSR=SR+40
970 IFS1=1THENSS=ES-100+EV
    :IFEV=60THENSS=SS+40
980 RETURN
```

When the FOR-NEXT loop is finished, the program prints the sunrise and sunset times previously calculated and ends at line 800. The program remains in a loop at line 800 so that the screen will not be disturbed by the usual ending routine of BASIC. You can press Shift-Break to exit the program.

Type in the program, changing line 175 to contain the latitude and longitude of your location. I scaled these off of a road map that had latitude and longitude along the edges. The coordinates must be in decimal form (no minutes or seconds).

If necessary, change lines 400 and 410 to fit your local time zone as described above. Before running the program, enter the date you want to run if it is not already in the machine. You just type DATE$="mm/dd/yy" to set the date.

If your applications for this program would require frequent changes of the date or coordinates then you may wish to modify the program to ask for these values as input from the keyboard. It is interesting to substitute a variety of coordinates and dates. You'll notice quite a variation in the path of the sun for different dates or latitudes. □

Industrial Strength Memory

By Alan L. Zeichick

An elephant may never forget, but a computer only remembers what it's told to remember. Well, up to certain limits, that is — unlike the human or elephant brain, computer data storage is finite, expensive and comparatively tiny.

If you need a lot of memory, and a disk drive's not appropriate for your environment, then you don't have too many choices. You have multiple-RAM banks, such as the new 224K RAM bank from P.G. Design Electronics, or the 512K bubble memory from SoundSight Magnetic Bubble Memory. And that's about it. Neither is inexpensive — but in many cases, cost isn't the most important concern.

On the surface — and legitimately, in most instances — the multiple-RAM bank solution is the best one. First of all, the P.G. Design 224K RAM bank, which gives a Model 100 a total of eight 32K RAM segments, costs $574. A 256K contiguous bubble memory costs $999 for the first 128K, plus another $450 for the memory upgrade, for a total of $1,449. If you're a Model 100 user who simply needs more memory, adding a multiple RAM bank or disk drive probably is the most cost-effective solution.

THINK AGAIN

But many situations require more than the RAM banks can offer: contiguous memory, rechargeable power for the memory and Model 100, and easy use from BASIC or machine-language. In these industrial-strength applications, what's a mere $1,449?

Of the many advantages that the bubble memory brings to the Model 100, two stand out. One is the fact that the memory is truly contiguous. In an unenhanced Model 100, there's a maximum of 32K of memory available for programs, data and BASIC workspace. With a multiple-RAM bank installed, such as P.G. Design's 224K bank, you don't have a nice, clean 256K — you have eight banks of 32K. That means that a program or data file can't grow to larger than 32K (actually, not even that large, since there's operating system overhead involved with each bank). Sorry, no massive 200K database, accessible with a 24K database program. Even with the new utilities which allow programs in one bank to access data in another, you don't have one straight line of memory like you have in a 256K desktop MS-DOS computer. That's the fault of the 80C85 processor, which can understand only 64K RAM through its address registers; and half of that, 32K, is the Model 100 operating system.

Enter the bubble. The bubble memory is a peripheral, not an extension of the system memory, connecting to the Model 100 through the system bus — the same way that the P.C.S.G. Chipmunk disk drive and the original Tandy Disk/Video Interface connected to the laptop computer. In just the same way that the Chipmunk can offer over 358K of contiguous random-access memory, so can the bubble offer 512K. (And, the computer can't even tell the difference!)

That value, 512K, is the maximum amount of memory currently available for the Model 100 in *any* form, disk or RAM, contiguous or in banks. *That's* worth something, if the application requires that much memory. How much is it worth? Well, the selling price is the same $999 for the first 128K, plus $450 per 128K expansion, or $2,349.

And, while we're comparing the memory potential of the bubble memory against a disk drive, bear in mind that each has its advantages. With a disk drive, you have virtually unlimited storage — off-line. That is, you can have hundreds of megabytes of data stored, but only 100K (on a Tandy drive) or 358K (on the Chipmunk) is available to you at one time. With the bubble, you have only 512K — period. So, if you're looking for a data-backup media, bubble isn't for you.

On the other hand, bubble memory is *fast* — nearly as fast as the CMOS RAM chips which provide main memory. If speed's an issue, a bubble memory is far superior to any mechanical device. And, an added feature: Since the bubble memory is a peripheral like a disk drive, instead of system memory like a multiple-bank RAM expansion, it retains its information even when the computer is cold-started. And, even longer: When *Portable 100* received a review bubble unit from SoundSight, it came completely loaded with software and sample data.

The elimination of moving parts gives the bubble other advantages, too: The unit can work in environments that would be unacceptable for a disk drive. For example, according to SoundSight, the bubble device can work in temperatures from -50 to 85 degrees Celsius, and can withstand 500 gravities of mechanical shock, compared to the 20 G limit for a mechanical disk drive.

PLUG AND PLAY

The SoundSight bubble memory device is nearly child's play to install. The device, which is about the same size as the Model 100, only about half as thick, attaches to the bottom of the laptop, and connects to it through the system bus via a seemingly fragile cable, on the same idea as the Chipmunk disk drive or the Tandy Disk/Video Interface. The bubble also contains its own power supply — in the review system, seven rechargeable batteries — which can be used to power the Model 100, too. If you own a Tandy 102, SoundSight will substitute a system bus cable that will fit your computer. This is one instance where the Model 100 is superior to its younger sibling, since the Tandy 102

cable will be exposed instead of fully enclosed by the bubble device.

The installation instructions comprise one and a half pages of the bubble's owner's manual, not including two diagrams. I would have been happier if the manual had more diagrams, especially for installing a microclip to the Model 100 battery compartment. The verbal instructions were vague, and the diagram wasn't very helpful.

Once the bubble memory is installed, the user is instructed to remove all machine-language .CO files from Model 100 RAM, and then do the ever-popular CALL 63012 from BASIC. This instructs the EPROM (yes, there's an option ROM included, and necessary) to create an invisible .CO file and a system file called *Bubble* in RAM. (Calling 63012 consumed 1,622 bytes of RAM.) The bubble program looks, and acts, like one of the Tandy portable disk drive operating systems: A Model 100-like menu displays five screens, or pages, each containing a maximum of 19 files. Don't confuse the bubble memory pages with RAM banks. The files are all stored in one location — just the file *names* can be grouped to enhance the organization of the 95 possible files.

Function key one, *V/RAM* or *V/bub*, toggles the screen from a RAM file menu to a bubble file menu. This allows the bubble program to operate on Model 100 main RAM files; pressing *F1* toggles between bubble and RAM. When viewing the bubble files, pressing the digit keys *1* through *5* shows the files in that respective memory page.

Function key two, *S/fl* or *L/fl*, moves from to and from the bubble memory, depending whether the bubble or RAM menu is displayed. Unlike many of the portable disk drive operating systems, the software doesn't offer a chance to change a file's name during a file transfer. A separate file renaming function must be performed.

A nice feature: The bubble memory directory is broken up into five pages. For example, to save a file TEST.DO from RAM into bubble directory two, run the bubble program, push *F1* to display the RAM menu, move the wide-bar cursor to TEST.DO, press *F2* for S/fl, then one of the digit keys *1* through *5*. The save operation takes only a couple of seconds, during which time a light-emitting diode (LED) on the bubble device lights up and the bubble memory buzzes very quietly.

Function keys five and seven, *K/fl* and *Rnam*, work on both RAM and bubble files. *F5*, which kills or erases files, flashes a "Are you sure?" message at the bottom of the screen; pressing any key except *Y* cancels the operation. *F7*, file rename, asks "Enter new file name," after which the file's name is set to whatever you chose.

The program should have more error checking. I attempted to rename a RAM file HI.DO to HI.BA, to see if the bubble program would refuse the operation. But the procedure completed — leaving me with a file HI.BA.DO. Evidently, the Model 100 text editor doesn't care about file names, since it allowed me to edit and update HI.BA.DO. BASIC refused to load or run the file, however, giving me a *?NM* error. Similarly, I was able to end up with multiple files within a bubble page with the same name, but not in RAM —the bubble program gave me a "File already exists" message in the latter situation.

UNMITIGATED GREED

After evaluating of the multitude of portable disk drive operating systems, I felt that the bubble operating system should have offered more. Just a few samples: Dumping multiple files to the bubble through a mark/copy operation; displaying each bubble and RAM file size; duplicating bubble files within the bubble for internal backup purposes and downloading from TELCOM directly into bubble memory (not possible — I tried). Maybe since the bubble memory ties up both the system expansion bus *and* the option ROM socket, I wanted enough to compensate me for the lack of ability to use ROMs such as Cleuseau, Ultimate ROM II, Super-ROM and Disk Plus, not to mention the Chipmunk disk drive.

While on that subject, I should mention that the portable disk drive seems to be compatible with bubble memory — as long as both aren't used simultaneously. I loaded TS-DOS into my Model 100, and was able to use that software to copy files to and from the portable disk drive, then execute the bubble program to talk to the bubble memory device.

The only conflict occurred in the area of RAM hooks for BASIC. Like all of the new portable disk drive operating systems, the bubble program allows you to run, load and save files, and use sequential file accessing to the disk drive (as device "B:"). It does this by taking advantage of the hooks, or RAM memory addresses, accessed by the Model 100 system ROM. So, from BASIC you can open "B:DATA.DO" for output as device number one, and write hundreds of thousands of bytes of data to a single file.

However, neither the bubble nor disk drive operating systems acknowledge each other's existence. I tried loading several operating system after the bubble program, and depending on which program was run last, I could access data files on either the disk drive or the bubble. As far as using both devices in the same program — forget it. It's not a hardware problem, since BASIC can read and write to the modem or serial port when the bubble's in use. But for the present, there's no software capable of using both the bubble and a disk drive.

But the bubble offers something that the portable disk drive does not: real random-access through program control. With the portable disk drive, the programmer only has sequential access, which means that a file that's opened for input can be read only top-to-bottom, and a file that's opened for output or append can only be added to, and only at the bottom. You can't change the third record of a 20-item database without reading and rewriting the entire file.

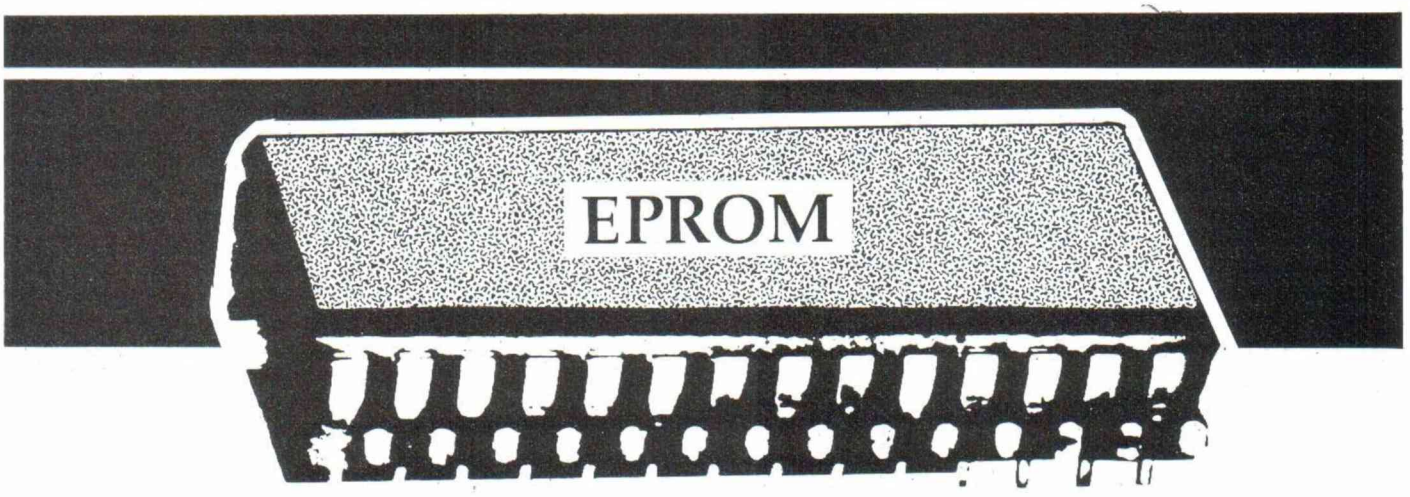
But with the bubble you can have true random-access manipulation of the memory — but it's only worth attempting if you're an experienced programmer. It requires calls to the bubble's EPROM, and a fixed record length of 254 characters. It's a bit of work, since *your* BASIC program is responsible for keeping track of pointers, telling the bubble's file-allocation table that a given sector is in use, and checking the table before using a sector. It's not difficult, if you're used to the principles involved with unidirectional linked lists, and are careful about debugging. The danger: Since random-access programming circumvents the bubble's file-management system, an erroneous or bug-ridden program can destroy or damage data or program files, requiring that the bubble be reinitialized. Fortunately, the reinitialization program is in the bubble ROM.

SoundSight makes no bones about it: The bubble memory is *not* for the casual computer user. It's a business tool, used by the government, large corporations like Standard Oil of Ohio, and special applications. It's an excellent and unique tool — no question. Is it for you? Only you, perhaps after consultation with SoundSight, can tell. □

ROM It Yourself

Burning your own ROM isn't just possible — but also affordable.

By Carl Oppedahl



In the early days of the Model 100, commercial programs, like home-written BASIC programs, were distributed on cassette tape. Later, both hobbyists and professional programmers began to use diskettes — both the 5.25-inch variety used by the Disk/Video Interface, and the smaller 3.5-inch Chipmunk and Portable Disk Drive variety. Now, many commercial products, such as SuperROM, Guardian, Sardine, Lucid and Ultimate ROM II, are to be found on EPROMs, or *eraseable programmable read-only memories.* But despite services offered by companies such as Traveling Software, Portable Computer Support Group and Polar Engineering, storing non-commercial software on an option ROM has been either unfeasible or just plain impossible.

Enter Radoyle. Equipment from the San Diego-based firm provides the hardware and software needed to create both custom option ROMs as well as replacements for the Model 100 operating system ROM.

WHAT'S IN A ROM

Before you can store your own programs on an EPROM, first you'll need to obtain one and make it work in the Model 100 family. Finding a blank 32-kilobyte EPROM is no problem; the commercially available EPROM that provides the nearest fit to the Model 100 (the 27C256) costs only a few dollars.

But it's not that easy to burn the EPROM itself. One is that specialized circuitry and voltages (not present in the Model 100) are needed to put binary information onto the EPROM. The device that actually performs this function is known, appropriately enough, as a *EPROM burner;* many vendors make models that can handle the 27C256. Another reason is that the Model 100's non-standard option ROM socket pin arrangement isn't friendly to those commercially available EPROMs.

Traveling Software, Polar Engineering and Portable Computer Support Group offer option ROMs for the Model 100. Each company has its own method for adapting commercially available EPROMs for use in the option ROM socket. Radoyle uses its own method, embodied in its two products, the ROM-OPT and ROM-STD modules.

Radoyle's ROM-STD is a kit that allows an EPROM that you've burned (i.e., programmed) to be used in place of the operating system ROM. ROM-OPT kit allows an EPROM to be used in the option ROM socket.

In each case, a printed circuit board about two inches square provides the necessary arrangement of signals. The EPROM sits above and off to one side of where it would be, had it been plugged directly into the socket.

That takes care of the pin-arrangement problem in the Model 100. But how do you burn the EPROM? One choice is Radoyle's TCJ-28, a test clip and jumper that lets you use nearly any commercial EPROM burner

Text continues on page 60

Powerful New Programs for your Radio Shack Model 100/102/200

DO NOT ADJUST YOUR SCREEN...

... nothing is wrong with your Model 100/102 that the new ULTRASCREEN™ program won't cure.

Normally, the Radio Shack™ Model 100/102 offers a screen with only a 320-character capacity - 40 characters per line, by 8 lines. Furthermore, if a word is too big to fit at the end of a line, the built-in Text editor shifts the ENTIRE WORD to the next line (this is called "word wrapping"). While this makes the text easier to read (no split words), too much screen space is WASTED at the right of almost every line.

Now, ULTRASCREEN gives you the power of a 60-character by 10-line screen, thereby correcting the main design flaw of your Model 100/102 - the small screen. With no decrease in the legibility of the characters, ULTRASCREEN more than doubles the effective viewing capacity of the 100/102: it increases the total number of characters in the display from 320 to 600, and reduces the number of word wraps by up to 1/3!

```
Additional features of ULTRASCREEN 100/1
02:◀
- Written entirely in machine language,
for speed of operation and minimum RAM
usage (less than 2.6K).◀
- Relocatable in memory - no problem to
use in conjunction with other machine
language programs.◀
```

```
Additional Features of ULTRASCREEN 100/102:◀
- Written entirely in machine language, for speed of
operation and minimum RAM usage (less than 2.6K).◀
- Relocatable in memory - no problem to use in conjunction
with other machine language programs.◀
- Full Model 100/102 character set is supported, including
GRAPHICS and INTERNATIONAL CHARACTERS.◀
- Can be used from all built-in programs (eg. BASIC, TEXT,
TELCOM, etc.), as well as your own BASIC or M/L programs.◀
- ESPECIALLY SUITED FOR USE WITH TEXT POWER 100/102.◀
```

FOR CATALOGUE, WRITE:

ULTRASOFT INNOVATIONS, INC.
76 Main Street / P.O. Box 247
Champlain, NY 12919-0247
ORDER LINES:
(800) 448-4511 (U.S. except N.Y.)
(800) 962-1480 (N.Y. state)
(800) 255-1133 (Ont. and Que.)
Ask for Extension 313-18
For product information, call:
(514) 457-9293

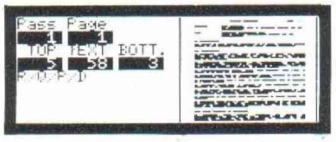
Orders shipped within 7 days of receipt (you will be informed of any undue delay). Items ordered together are not necessarily shipped the same day. TEXT POWER, PAGE PLOT and MERGE TEXT are registered trademarks of Hugo Ferreyra.

YOU DIDN'T GET WHAT YOU THOUGHT YOU WERE GETTING, DID YOU?

If you bought a Model 100/102 for word processing, you were looking for CONVENIENCE, FLEXIBILITY and PORTABILITY. The problem was, the built-in TEXT program was only HALF a word-processor - it allowed you to enter the data into your computer and edit it, but didn't help you at all to set up that page in an attractive format for printing.

Now, TEXT POWER™, "the best formatter there is for the Model 100", according to reviews in both PORTABLE 100 and CREATIVE COMPUTER magazines, is the key which will unleash the capacity of your 100/102 to give you:

- the POWER to **create or edit text files, and save them to disk or cassette**, without ever leaving the program,
- the POWER to **format text automatically**, with tremendous speed,
- the POWER to **use all the familiar built-in TEXT features,** such as PASTE, CUT, COPY, etc.
- the POWER to **insert text at specified locations in the master text file,** using MERGE TEXT™ (e.g. insert names and addresses in a business letter),
- the POWER to **add headers and footers**, which can be turned on or off (or changed) from within the text,
- the POWER to **right-justify text,** and/or to **center text vertically or horizontally,**
- the POWER to **control left and right margins for the entire text, or for individual paragraphs,**
- the POWER to **number pages automatically in any location,** and
- the POWER to **preview the complete layout of your formatted text graphically, without embedded commands, before printing** (using PAGE PLOT™:



Each version of TEXT POWER is customized for a particular printer, or group of printers, allowing the same commands to be used, even if the actual printer code varies from one machine to another. For example, you can use the following features (individually or in combination, if your particular printer supports them): boldface, condensed, double-strike, elite type, near-letter-quality (NLQ), superscript and subscript, italics, underlining, etc.
TEXT POWER also gives you the ability to send automatic line feeds to your printer, and to advance the paper to the top of the next page, from within the text.
Optional support is available for INTERNATIONAL CHARACTER SETS - including French, Spanish, German, etc. - for use with virtually all printers.

YES! PLEASE SEND ME THE PRODUCTS INDICATED AT RIGHT.

NAME
ADDR.
CITY
STATE ZIP
TEL. (DAY)
PAYMENT BY: ☐ CHK/MO ☐ C.O.D. ☐ VISA/ ☐ MC
EXP.
SIGNED

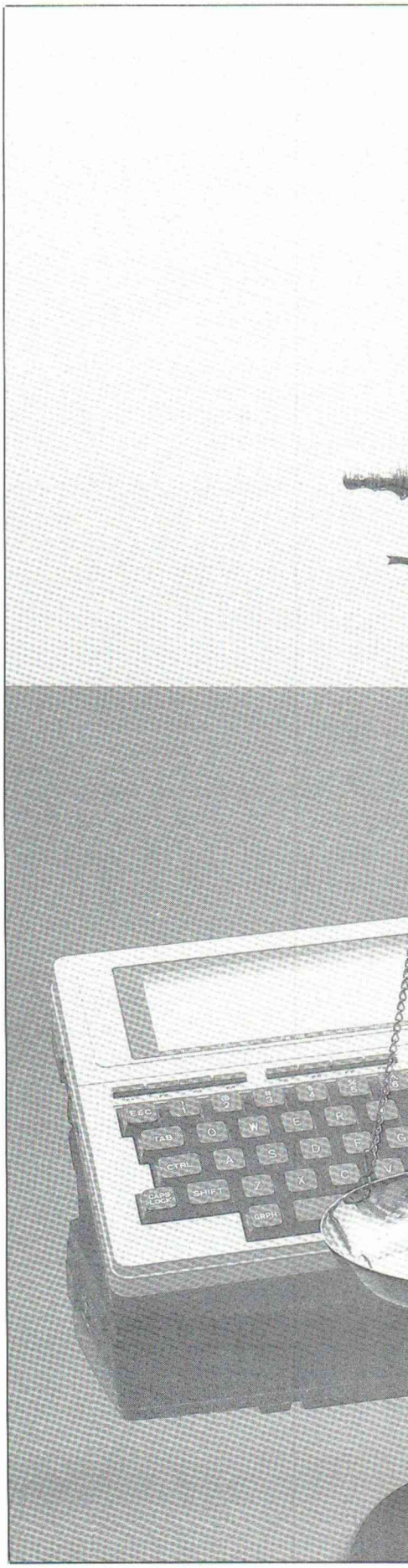
QTY	DESCRIPTION	PRICE	TTL
	ULTRASCREEN	$34.95	
	TEXT POWER	$59.95	
	8K CHIPS-100	$22.95	
	8K CHIPS-102	$ 9.95	
MODEL: ☐ 100/ ☐ 102			
TEXT POWER ORDERS: Printer: ☐ Parallel/ ☐ Serial			
	SUBTOTAL		
	SHIPPING/HANDLING		4.00
	C.O.D. ($3.00)		
	TOTAL		

Circle 78 on Reader Service Card

It's only a small percentage of the Tandy 102's system ROM that's different from the Model 100. But even these small changes are significant.

Weighing the Differences

By Carl Oppedahl



The press release announcing the new Tandy 102 touts it as "software and hardware" compatible with the venerable Model 100, but there are significant, albeit small, software differences between the two.

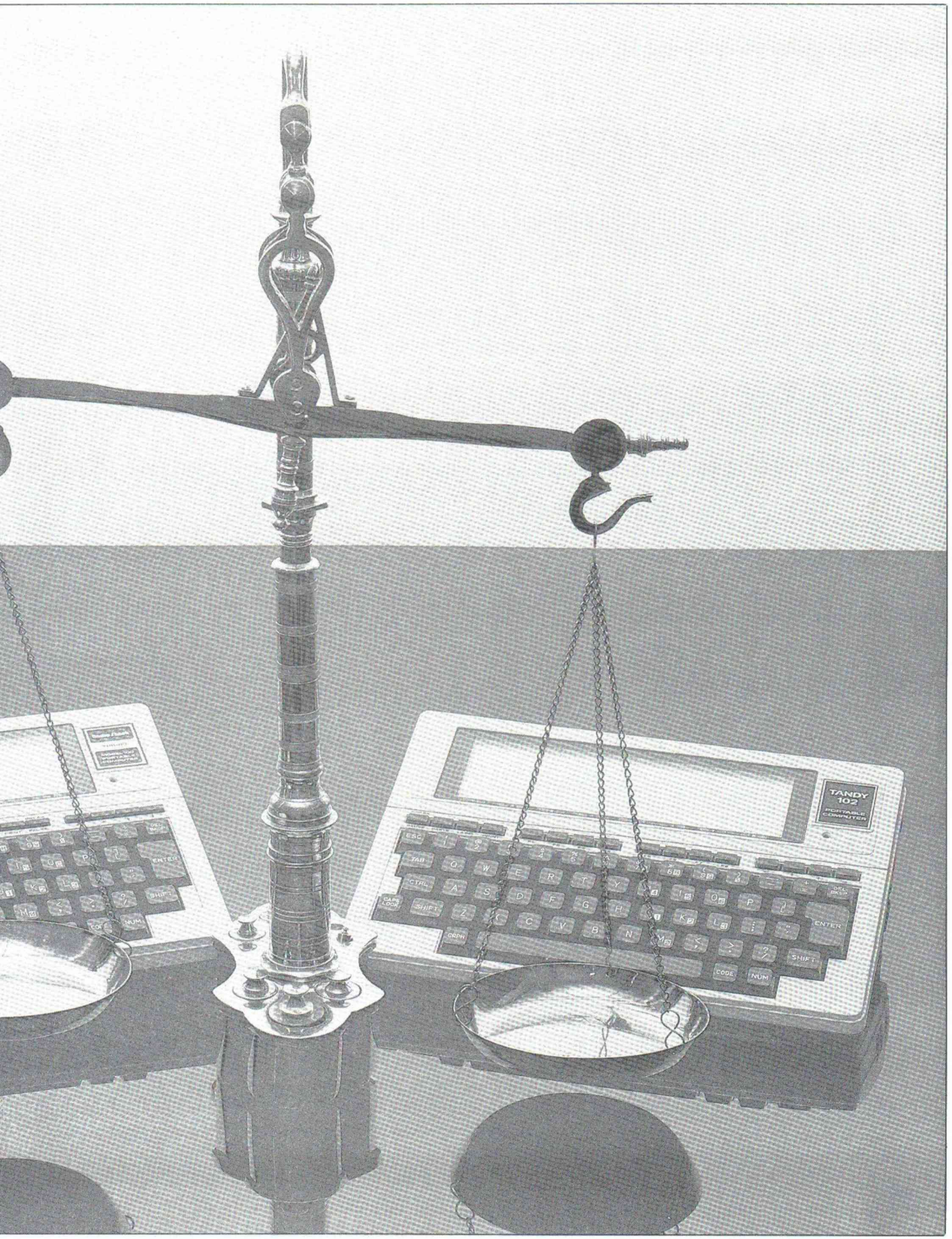
To write this article I made an exhaustive, byte-by-byte comparison and analysis of the ROM operating system and applications software of the Tandy 102 and Model 100. I disassembled and studied the 1.09 percent of ROM that differed between the two machines, and my findings are discussed here.

Microsoft made changes in the areas of ROM having to do with keyboard decoding and LCD character generation. For example, the INKEY$ values generated by Code key sequences are different. To see this, run the following program:

```
1 I$=INKEY$:IF LEN(I$) THEN PRINT ASC(I$)
2 GOTO 1
```

The program does nothing until you push a key that is meaningful to the computer. For example, if you push "A" (Shift and "a") the number 65 is displayed, showing that the ASCII value for capital A is 65. If you push Shift (or Ctrl, or Grph, or Code) no number will be displayed, because none of those keys has an ASCII value. (They simply modify the value of some other key, such as a letter of the alphabet.) Each letter key by itself has an ASCII value, but some key combinations, such as Grph-D, do not have a value. As far as the computer is concerned, pushing Grph-D is like not even touching the keyboard.

This leads to the first major difference between the 100 and 102: Certain Code key combinations that are meaningful (yield an INKEY$ value) in the 100 are meaningless in the 102, and vice versa. Model 100 programs that depend on Code key input or display of Code characters, will require modification before they can run on the 102.


TANDY
102
PORTABLE
COMPUTER

DIFFERENCES

There are 10 combinations that are meaningful on the 100 but not on the 102, and 10 others that are meaningful on the 102 but not on the 100. In addition, there are 14 combinations that are meaningful on both computers but yield differing ASCII values.

The ASCII values for which there are new Code combinations fall between lines 161 and 223. In the Model 100 any desired ASCII value could be generated with a properly chosen key combination, and this remains the case in the 102. What has changed is the particular key combination to use for certain ASCII values. The portion of the 102 ROM that brings about these changes is between location 7CA1 and 7CF3 hex.

DISPLAYED LCD CHARACTERS

There is also a completely independent way that characters have changed in the Tandy 102, and that is in the displayed character for selected ASCII values. For example, the 102 is able to display 10 characters the 100 cannot; these were apparently designed to permit use of the Tandy 102 in Scandinavian and Spanish-speaking countries. The ROM changes to bring this about are in memory location 77BB to 7B2F.

The ROM region that defines the ASCII values displayable on the LCD was also modified to enhance the shape and readability of some characters, including the shape of the letters *B, C, D, p, q, s,* and *u*. Each displayed character is composed of forty-eight pixels, six across and eight tall, and the ROM determines which ones are dark for a given ASCII value. (This is described in chapter 13 of *Inside the Model 100*.) The ROM pixel templates have been changed for the following values: 66-68, 112, 113, 115, 117, 160, 162, 164, 165, 169, 172-175, 177-179, 181, 183, 185, 190-195, 197, 199, 200, 201, 206, 207, 216-218, and 223.

Byte-by-byte comparison of the corresponding regions of the Tandy 200 ROM reveal that the Code keyboard values and enhanced LCD pixel maps of the 102 are identical to those of the 200. This means, ironically, that BASIC programs written for the 102 that rely on Code characters at the keyboard or screen will run better on the 200 than on the 100.

The two major categories of change in the 102 (the key-combination changes and the LCD pixel-map changes) can interact with each other, and this shows up most prominently in text. A keystroke works its way to the screen in two steps: First the key combination is converted to a numerical ASCII value, and then the ASCII value is translated to a pixel pattern on the screen. If you put a 100 and a 102 side by side, and type into text files in the two machines, you will from time to time encounter a keystroke that displays one way on the 100 and differently on the 102. The important thing to realize is that the difference may arise due to changes in the keystroke-to-number routine or the number-to-screen routine.

Here's an example of the former. If you're in a text file in the 100 and push CODE-a, you will see on the screen an "a" with umlaut. In the 102 a Code-A gets you an "a" with an ague accent. The displayed characters differ because Code-A means an ASCII value of 182 in the 100 but means an ASCII value of 200 in the 102. In either computer, 182 means the "a" with an umlaut and 200 means the "a" with the ague accent. You can see this by printing CHR$(182) and CHR$(200) within BASIC in the two machines.

Here's an example of a change in the number-to-screen routine. With the two computers side by side in text, push Grph-1. The 100 will show you an inverted exclamation point, while the 102 will show you a Greek pi symbol. In either computer, a Grph-1 is treated as an ASCII value of 136. The differing display is due to the LCD pixel map assigned to the value 136.

If you have the computer hooked up to a printer, or are telecommunicating, it's important to understand the difference between the two situations. If you have typed a Grph-1 into text files in both a 100 and a 102, the two files will be identical when printed on paper even though they differ on the LCD screen.

Most time/date information in the 100 (and 102) is kept and updated by the clock chip in its own RAM, but the year is maintained in ordinary RAM by the operating system. Model 100 owners have often seen the year advance, so that what was 1986 this morning becomes 1989 by this afternoon. What the 100 operating system tries to do is check to see if the month has changed, and if it has the year gets incremented. The problem was that the routine was thrown off by interrupts and would sometimes get mistaken about the month, and spuriously update the year.

In the 102 the year-update routine

CODE	Model 100 value	Tandy 102/200 value
,	188	*
.	207	213
a	182	200
b	*	207
c	162	189
f	*	162
m	*	188
q	200	182
v	189	206
w	*	211
x	161	209
y	204	*
z	206	161
<	221	*
>	*	212
?	*	220
!	208	204
#	209	*
&	212	*
*	210	*
(	211	*
)	*	165
A	177	216
C	171	222
E	214	*
I	213	*
M	165	221
N	*	214
Q	216	177
V	222	*
W	*	210
X	223	208
Y	220	171
Z	*	223

** means no ASCII value is produced by the key combination*

(at location 1B8C to 1BAA hex) has simply been eliminated.

The calendar functions have changed in one other way: the abbreviation for *July* appearing at the main menu is displayed as *Jly* on the Model 100, while in the 102 it's *Jul*. This was done by changing the month table at 5AFC.

The default telcom stat setting (the one you get when the machine is new or cold-started) has been changed from M7I1E to M7I1D, disabling the XON/XOFF protocol. I expect this was done to help users who would be confused by the locked-up keyboard that results when the distant computer sends a Ctrl-S. The ROM was changed at location 03C9 for this. You can still manually enable or disable XON/XOFF — only the default has changed.

Telcom has also been changed (at 54CD and 7315 to 7326) so that in modem mode (as distinguished from RS-232 mode) the 102 ignores any delete character received from the distant computer. The 100 treated a received delete character just as it did a backspace, and so does the 102 when in RS-232 mode.

ASCII value	Model 100 code key	Tandy 102/200 code key
161	x	z
162	c	f
165	M	]
171	C	Y
177	A	Q
182	a	q
188	,	m
189	v	c
200	q	a
204	y	!
206	z	v
207	.	b
208	!	X
209	#	x
210	*	W
211	(	w
212	&	>
213	I	.
214	E	N
216	Q	A
220	Y	?
221	<	M
222	V	C
223	X	Z

Code key combinations

The value of Pi was incorrect in the Model 100, from the tenth decimal place on. This has been corrected in the 102, at ROM location 33BF. Small changes have been made to the power-on routine, to PRINTUSING, to the text Find function and to the cassette Skip: routine.

Because of the differences between the 100 and 102, you may want your BASIC program to be able to figure out which model of computer it's running in. One good way is by peeking the contents of the second and third locations in ROM.

It was quite an engineering feat to squeeze all the familiar Model 100 software into a mere 32K of ROM, and nearly every byte was spoken for. Where did Microsoft find room for the added bytes?

Some of the new code landed on top of Suzuki, Hayashi, and Ricky, the never-displayed filenames for such things as the paste buffer and the unnamed BASIC program. Some of it was shoehorned in where the calendar year-update routine was. There were six unused bytes at the very end of the Model 100 ROM, and they have been called into service. One Model 100 routine at 7304 that I could see no use for (it looked like a routine for a system-bus disk drive) has been gutted in the 102 to make room for the PRINTUSING patch and the Telcommunications patch.

Yes, the Tandy 102 is nearly "software and hardware" compatible with the 100. Tandy's introduction of the 102 assures the continued health of the laptop market by extending the useful life of the vast world of Model 100 software. Virtually any machine language or BASIC program written for the 100 will work fine on the 102. If, however, a program relies on Code keys or characters, you may have to modify the program before running it on the 102.□

ASCII value	Model 100 displays	Tandy 102 displays
136	ì	π
204	ý	ì
208	Â	Æ
209	Ê	æ
210	Î	À
211	Ô	à
212	Û	Ø
213	Ï	ø
214	Ë	N̄
220	Y	¿

Changed ASCII values between Model 100 and Tandy 102

Computer	PEEK(1)	PEEK(2)
Model 100	51	125
Tandy 102	167	27
Tandy 200	171	152

By PEEKing the second and third memory locations you can determine which computer you are working with.

Time to Program

Taking a walk through the program-creation process.

By Alan L. Zeichick

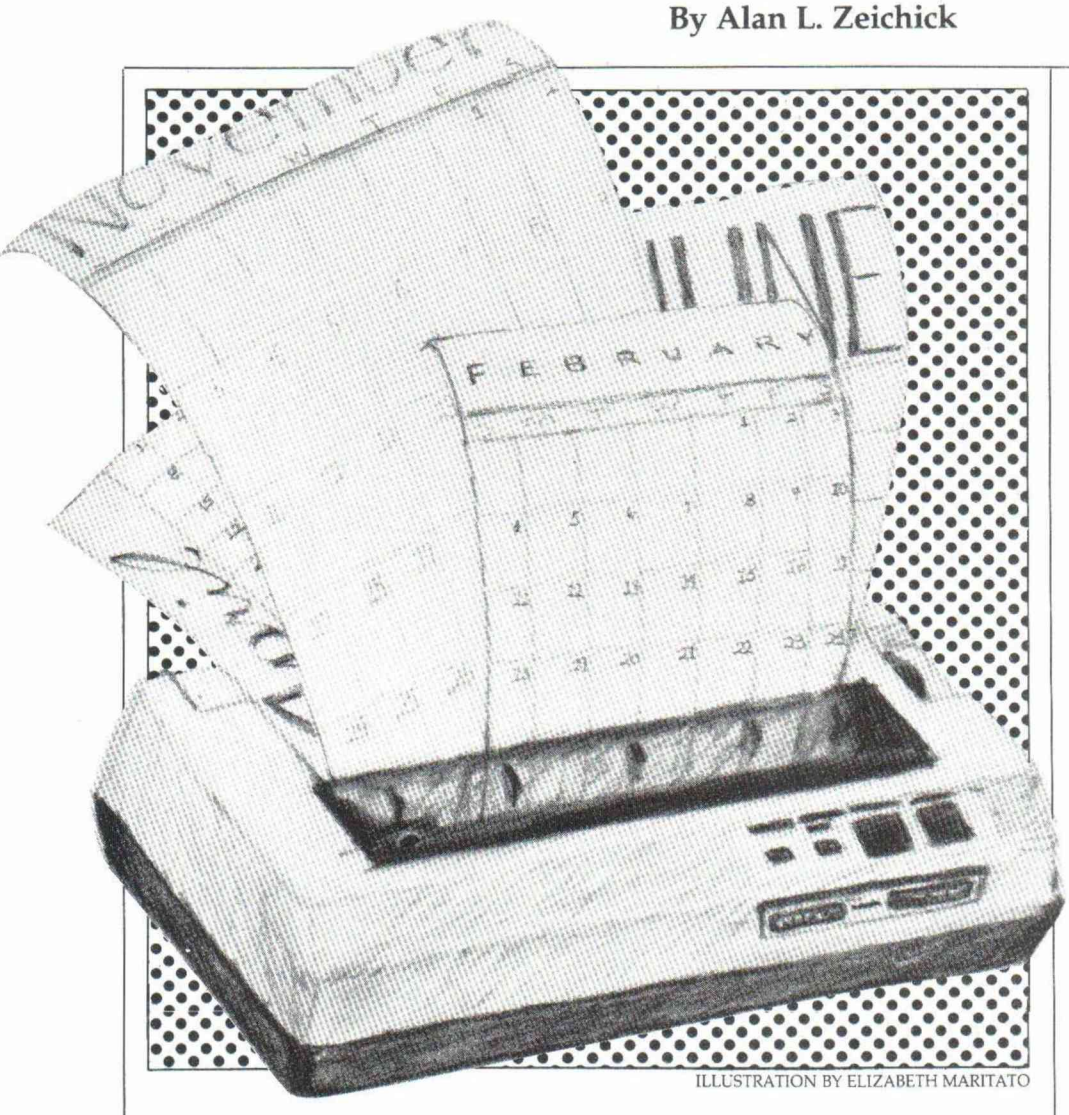


ILLUSTRATION BY ELIZABETH MARITATO

Computer programming has nothing to do with BASIC. Or machine language, C, or even the computer. Programming is a *discipline,* a method of solving problems. A computer programmer breaks a task down into individual steps, each of which can be directly translated into computer-understandable instructions.

All of us, from time to time, have used datebooks to jot down anniversaries, lunch date or business appointments. The simplest datebook, of course, is a plain old calendar, with space near each day for printing memos and reminders.

So let's use our laptop's BASIC to manufacture blank datebook calendar pages.

Were we in the science-fiction world, we could just type on our Tandy 20,000 keyboard, "Print off calendars for the year 2,040." But we're not, and "print calendar" isn't a computer-understandable command (unless someone's already given you such a program). That's where the programmer comes in, breaking down the problem (making a datebook calendar) into smaller, easier-to-digest portions.

Of course, a programmer *does* have to know what portions the computer can handle — and that's where his or her knowledge of a programming language (such as BASIC) comes in handy. But even if you're not fluent in BASIC, keep reading. With a copy of your BASIC manual, you should have no trouble understanding our simple program.

BREAKING UP ISN'T HARD TO DO

Programming is just like baking a cake. In fact, programs are often compared to recipes.

The first step towards baking a calendar is to define the major tasks that we must perform. Here they are, in the order in which we must perform them:

- Decide which month and year to start with, and how many months are needed.
- For each month:
- Calculate the starting day of the week and number of days in that month.
- On a new sheet of paper, print the month and year, and the names of the days of the week.
- For each week (Sunday through Saturday) print boxes for each day.

The first part looks pretty easy. Here's a rough sequence of statements that will fulfill those requirements. I'm ignoring the two-letter variable name restriction that laptop BASIC imposes; when we get ready to type in the listing, I'll change the long, meaningful names to some clever mnemonics.

```
1000 PRINT "What's the starting
     month and year?"
1010 PRINT "Please separate them
     with a comma."
1020 INPUT MONTH, YEAR
1030 IF YEAR < 100 THEN YEAR =
     YEAR + 1900
1040 PRINT "How many calendars
     should I make?"
1050 INPUT NUMBER
```

Line 1030 is one that many new programmers wouldn't think of. Yes, it might be easier to prompt the user to type all four digits of the year, but why make more work for the user? Let the computer figure it out.

(A truly user-friendly program should be able to figure out that "May 1986," "6/86" and "6-1986" are all the same thing. We'll leave that capability as an exercise to the reader.)

This part of the program should also check the validity of the input, rejecting months numbered 15, for example. The full program listing will contain more of the data-checking statements.

The second code part of our breakdown isn't too difficult:

2000 FOR LOOP = 1 TO NUMBER

For those unfamiliar with the FOR-NEXT statement, this line instructs the computer to repeat a section of program code several times. The first time the section of code is executed, the variable LOOP will have the value 1; the second time, it will be equal to 2. This will continue until LOOP is equal to NUMBER, the number of months the user requested. The block of code to be repeated will include every line from line 2000 until the statement NEXT LOOP.

PICK A DAY, ANY DAY

Now, for a difficult process: finding which day of the week our input month starts on. Most date calculations use pretty intricate formulas, and this one is no exception.

The date-to-day formula is common. I plucked it out of a scheduling program I wrote several years ago, but it can be found in many computer reference books.

Here's an algorithm for determining upon which day of the week a given day falls:

IF MONTH > 2 THEN WEEKDAY = 365*YEAR + DAY + 31*(MONTH - 1) - INT (0.4*MONTH + 2.3) + INT(YEAR/4) - INT (¾*INT((YEAR/100) + 1))

IF MONTH < 3 THEN WEEKDAY = 365*YEAR + DAY + 31*(MONTH - 1) + INT ((YEAR - 1)/4) - (INT (¾*INT(((YEAR - 1)/100) + 1)))

WEEKDAY = WEEKDAY - INT (WEEKDAY/7)*7

CALBA: A program for generating monthly calendars.

```
1 'Calendar Program
2 'Alan L. Zeichick, Portable 100
3 '
4 ' Initialize the Arrays
5 '
10 DIM DN$(7), MN$(12), MS(12)
20 FOR CT = 1 TO 7
30 READ DN$(CT)
40 NEXT CT
50 DATA Sunday, Monday, Tuesday, Wednesday,
   Thursday, Friday, Saturday
51 '
60 FOR CT = 1 TO 12
70   READ MN$(CT), MS(CT)
80   NEXT CT
90 DATA January, 31, February, 28, March, 31,
   April, 30, May, 31, June, 30, July, 31, August,
   31, September, 30, October, 31, November, 30,
   December, 31
91 '
92 ' Get the date and number of months
93 '
1000 PRINT "What's the starting month and year?"
1010 PRINT "Please separate them with a comma."
1020 INPUT MM, YY
1030 IF YY < 100 THEN YY = YY + 1900
1040 PRINT "How many calenders should I make?"
1050 INPUT NM
1051 '
2000 FOR LP = 1 TO NM
2001 '
2002 ' Calculate starting date
2003 '
3000   IF MM > 2 THEN WD = 365*YY + 1 + 31*(MM-1) -
       INT(.4*MM + 2.3) + INT(YY/4) - INT(3/4*INT
       ((YY/100) + 1)) ELSE WD = 365*YY + 1 +
       31*(MM - 1) + INT((YY - 1)/4) -(INT(3/4*INT
       (((YY - 1)/100) + 1)))
3010   WD = WD - INT(WD/7)*7
3020   IF WD < 1 THEN WD = 7
3030   DY = MS(MM)
3040   IF MM = 2 AND YY MOD 4 = 0 AND YY MOD 100 <>
       0 THEN DY = 29
3041 '
3042 ' Print the month name
3043 '
4000   TT$ = MN$(MM) + "," + STR$(YY)
4010   SP =(71 - LEN(TT$))/2 + 10
4020   LPRINT SPACE$(SP); TT$
4030   LPRINT
4040   LPRINT
4041 '
4042 ' Print day names
4043 '
4050   FOR DC = 1 TO 7
4060     SP =(9 - LEN(DN$(DC)))/2 + 1 + DC*10
4070     LPRINT TAB(SP); DN$(DC);
4080     NEXT DC
4090   LPRINT
4091 '
```

A WEEKDAY value of 1 indicates Sunday, 2 is Monday, and 7 is Saturday. In our program, we're always going to start on the first of the month, so we can exchange the variable DAY for the constant 1:

```
3000 IF MONTH > 2 THEN
     WEEKDAY = 365*YEAR + 1 +
     31*(MONTH-1) - INT
     (0.4*MONTH + 2.3) +
     INT(YEAR/4) -INT (3/4*
     INT((YEAR/100) + 1)) ELSE
     WEEKDAY = 365*YEAR + 1 +
     31*(MONTH - 1) + INT ((YEAR -
     1)/4) -(INT (3/4*INT(((YEAR -
     1)/100) + 1)))
3010 WEEKDAY = WEEKDAY - INT
     (WEEKDAY/7)*7
3020 IF WEEKDAY < 1 THEN
     WEEKDAY = 7
```

The third part of our problem breakdown also asks that we determine the number of days in each month. Microsoft BASIC doesn't have a built-in function to do that, and it's never heard the childhood rhyme, "Thirty days has...."

Let's add some statements at the top of the program to predefine, or *initialize,* an array for the number of days in each month. While we're at it, let's teach it the days of the week and the names of the months.

```
10 DIM DAYNAME$(7),
   MONTHNAME$(12),
   MONTHSIZE(12)
20 FOR COUNT = 1 TO 7
30 READ DAYNAME$(COUNT)
40 NEXT COUNT
50 DATA Sunday, Monday, Tuesday,
   Wednesday, Thursday, Friday,
   Saturday
60 FOR COUNT = 1 TO 12
70 READ MONTHNAME$(COUNT),
   MONTHSIZE(COUNT)
80 NEXT COUNT
90 DATA January, 31, February, 28,
   March, 31, April, 30, May, 31, June,
   30, July, 31, August, 31, September,
   30, October, 31, November, 30,
   December, 31
```

The first statement, DIM, stands for *dimension;* it tells the computer that we're setting up three arrays, one for seven character strings, one for 12 character strings, and one for 12 numbers.

Think of an array as a bunch of post-office boxes, and the names of the arrays as individual post offices. The variable DAYNAME$(3) refers to the P.O. Box 3 in the town of DAYNAME$. By using these references in programs, and by changing the P.O. Box number (called the *index*), we have a powerful way of storing and accessing information.

By the way, according to the rules of Microsoft BASIC, we didn't have to dimension DAYNAME$, since it has fewer than ten elements. I recommend that you dimension it anyway — it doesn't cost anything, and then you can identify all of your arrays simply by checking out the DIM statement.

BACK TO THE PROBLEM

We *still* have to determine the number of days in the month that we're printing. It's normally the number of days in the generic month — but watch out for leap years!

```
3030 DAYS = MONTHSIZE(MONTH)
3040 IF MONTH = 2 AND YEAR
     MOD 4 = 0 AND YEAR MOD 100
     <> 0 THEN DAYS = 29
```

A leap year is defined as one which is divisible by 4, but *not* divisible by 100. The MOD (modulo) function returns the remainder after division. What we're doing is defining three condi-

```
4092 ' Back up to Sunday and begin
4093 '
5000   SD = 2 - WD
5010   WK = INT(((MS(MM) + WD - 8)/7) + 2)
5011 '
5012 ' Repeat for each week in the month
5013 '
5020   FOR WC = 1 TO WK
5021 '
5022 ' Print the top dividing line
5023 '
5030     LPRINT SPACE$(10);
5040     FOR DC = 1 TO 7
5050       LPRINT "+----------";
5060       NEXT DC
5070       LPRINT "+"
5071 '
5072 ' Print the date line
5073 '
5080     LPRINT SPACE$(10);
5090     FOR DC = 1 TO 7
5100       LPRINT "|";
5110       IF SD >= 1 AND SD <= DY THEN LPRINT
           USING "#########"; SD; ELSE LPRINT
           "          ";
5120       SD = SD + 1
5130       NEXT DC
5140     LPRINT "|"
5141 '
5142 ' Print the day boxes
5143 '
5150     FOR BC = 1 TO 7
5160       LPRINT SPACE$(10);
5170       FOR DC = 1 TO 7
5180         LPRINT "|          ";
5190         NEXT DC
5200       LPRINT "|"
5210       NEXT BC
5211 '
5212 ' End the each-week loop
5213 '
5220     NEXT WC
5221 '
5222 ' Print the bottom dividing line
5223 '
5230   LPRINT SPACE$(10);
5240   FOR DC = 1 TO 7
5250     LPRINT "+----------";
5260     NEXT DC
5270   LPRINT "+"
5271 '
5272 ' Page eject, go to next month
5273 '
6000 LPRINT CHR$(12)
6010 MM = MM + 1
6020 IF MM > 12 THEN YY = YY + 1:MM = 12
6030 NEXT LP
6040 END
```

tions which have to be met to produce a 29-day month: it must be February, the year must be evenly divisible by four, and the year must not be evenly divisible by 100.

WEEK BY WEEK

The next part of the problem breakdown wants us to print the month and year at the top of a new sheet of paper. Let's assume that the last program executed prior to our calendar left us at the top of the page, and we'll end this month on a new sheet, too.

It would be nice to center the month and year on the page, but before we do that, let's figure out how wide our calendar will be.

Using 10-cpi (characters per inch), there are 85 characters across on a standard letter-size sheet of computer paper. But let's leave wide margins, and use only 71 characters. That's ten characters for each day of the week, plus one character to conclude each line. And, let's begin ten characters (one inch) from the left-hand margin.

The text we're going to print is should look like *May, 1986* and we can produce that by adding the month's name, a comma, and the character representation of the year together:

```
4000 TITLE$ =
     MONTHNAME$(MONTH) + ","
     + STR$(YEAR)
```

The built-in function STR$ (string) converts the internal binary-format number to ASCII-format characters before adding it to the character string.

Centering a string requires printing spaces in front of it. The length of the area we're centering on is 71 spaces; if we subtract the length of our title, we'll know how much blank space we have, and dividing that amount by two should yield the number of spaces we'll need to print. How do we find the length of the title? With the built-in function LEN.

```
4010 SPACES = (71 - LEN(TITLE$))/2 +
     10
```

Actually, we don't have to print spaces; we can use the TAB (tabulator) function to print to an exact column. Let's not forget to add the 10-character left-margin. Plus, when keying your program, don't leave a space between the word TAB and the following parenthesis; if you do, you'll get a error. Normally, such spaces are ignored, but not with this keyword.

```
4020 LPRINT TAB(SPACES); TITLE$
```

LPRINT is the BASIC statement used to output information to the physical printer, rather than to the liquid-crystal display. After this, let's skip a few lines on the paper:

```
4030 LPRINT
4040 LPRINT
```

Of the ten horizontal characters we're allotting to each day, the first will be a vertical line (we'll use a pipe, which is that character made up of two vertical lines) and the next nine will be blanks. Let's use a similar technique to center the names of the days within their boxes:

```
4050 FOR DAYCOUNT = 1 TO 7
4060 SPACES = (9 -
     LEN(DAYNAME$(DAYCOUNT)))
     /2 + 11 + DAYCOUNT*10
4070 LPRINT TAB(SPACES);
     DAYNAME$(DAYCOUNT);
4080 NEXT DAYCOUNT
4090 LPRINT
```

We're using a margin of 11 to compensate for the fact that we're ignoring the vertical-line part of the day field.

The semicolon at the end of the LPRINT statement tells BASIC not to advance to the next line on the page, so all of our weekday names will be on the same line. The final LPRINT, though, ends the sequence and jumps to a new line.

DRUM ROLL, PLEASE

Now comes the part we're all been waiting for: actually printing the calendar.

It makes sense to print the calendar week by week, since we're trying to match the standard calendar format. Although we're printing Sunday as the first day of the week, it's not always the first day of the month.

Therefore we need to leave blank days at the beginning of the month if the first isn't a Sunday. If we're printing an October calendar and the first day of the month is Tuesday, then Monday would be October 0 and Sunday would be October -1.

In line 5000, let's compute the date of that initial Sunday, the first day on our monthly calendar:

```
5000 STARTDAY = 2 - DAYS
```

Now we need to determine how many weeks we need to print. It's actually the number of weeks needed to cover all the days from the starting Sunday through the last day in the month. That is represented by MONTHSIZE(MONTH) - STARTDAY. The number of weeks is that figure divided by 7 — and rounded upwards:

```
5010 WEEKS = INT(((MONTHSIZE
     (MONTH) - STARTDAY) + 6)/7)
```

Variable abbreviations for CAL.BA

Mnemonic	Full Name	Use
BC	BLANKCOUNT	Loop index for each day's height
CT	COUNT	Loop index for reading DATA statements
DC	DAYCOUNT	Loop index for print each day's information
DN$	DAYNAME$	Array of the names of the days
DY	DAYS	Number of days in the current month
LP	LOOP	Loop index for each printed month
MM	MONTH	Current month (1-12)
MN	NUMBER	Number of months to make calendars for
MN$	MONTHNAME$	Array of the names of the months
MS	MONTHSIZE	Array of the lengths of the months
SD	STARTDAY	Beginning position for the month
SP	SPACE	Number of blanks before printing
TT$	TITLE$	Month, comma, year
WC	WEEKCOUNT	Keep track of weeks
WD	WEEKDAY	Weekday of the first of the month (1-7)
WK	WEEKS	Number of weeks in the month
YY	YEAR	Current year (1900-2100)

PROGRAMMING WORKSHOP

How much vertical space can we allot to each week? There are a maximum of six weeks in a month (using our above definition). A sheet of standard paper has 66 lines, but we don't want to use more than 60, and we've used four for the title. That leaves 56—and 54 is evenly divisible by six, yielding nine.

Each week, therefore, will use nine lines, with the first being a line of 71 minus signs. The second line will contain the day number, right-justified in the 9-character field, and the last seven will only contain the vertical lines which separate the days from each other.

Note that we'll only print the day number when the day is between the first and last days of the month.

First, put out the top dividing line:

```
5020 FOR WEEKCOUNT = 1 TO
     WEEKS
5030 LPRINT TAB(10);
5040 FOR DAYCOUNT = 1 TO 7
5050 LPRINT "+---------";
5060 NEXT DAYCOUNT
5070 LPRINT "+"
```

There are nine minus signs in the quoted expression on line 5050, following the plus sign. The plus sign is used to indicate the junction of the vertical and horizontal dividers. The final plus terminates the dividing line.

Next, print the line which contains the day numbers. (To type the pipe (vertical line) character, use the key sequence Grph-Shift-Minus.)

```
5080 LPRINT TAB(10);
5090 FOR DAYCOUNT = 1 TO 7
5100 LPRINT "|";
5110 IF STARTDAY >= 1 AND
     STARTDAY <= DAYS THEN
     LPRINT USING "#########";
     STARTDAY; ELSE LPRINT " ";
5120 STARTDAY = STARTDAY + 1
5130 NEXT DAYCOUNT
5140 LPRINT "|"
```

The USING clause of the LPRINT statement ensures that the day number will be right-justified in a nine-character field. The blank string in the ELSE clause, which is executed for those days outside of the month, simply prints nine spaces.

We decided to have nine lines for each week, and we've used two. So,

let's print seven almost-blank lines:

```
5140 FOR BLANKCOUNT = 1 TO 7
5150 LPRINT TAB(10);
5160 FOR DAYCOUNT = 1 TO 7
5170 LPRINT "|        ";
5180 NEXT DAYCOUNT
5190 LPRINT "|"
5200 NEXT BLANKCOUNT
```

Don't forget to indicate that we have more weeks to print:

```
5210 NEXT WEEKCOUNT
```

We're nearly done. To top this calendar off, we have to insert the closing line for the last week, and jump to a new page by printing the special ASCII code number 12, which means "top of form":

```
5220 LPRINT TAB(10);
5230 FOR DAYCOUNT = 1 TO 7
5240 LPRINT "+---------";
5250 NEXT DAYCOUNT
5260 LPRINT CHR$(12)
```

Finally, we have to advance the month by one, and prepare to repeat the entire process by closing the FOR statement we introduced at step 2.

```
6000 MONTH = MONTH + 1
6010 IF MONTH > 12 THEN YEAR =
     YEAR + 1:MONTH = 1
6020 NEXT LOOP
```

THE FINAL VERSION

If you assemble all of the lines used in the article, you have a complete, ready-to-use datebook program. I've only made two changes: adding some internal documents for the major part of the program, and abbreviating all the long variable names down to two letters.

Don't let any programmer kid you. Learning the programming language is the easiest part of programming, requiring that you study the manual. The hard part is conceptualizing the problem as a series of very definite sequential operations. Certainly, on the whole any task can seem overwhelming — be it developing a multi-user accounting system or only printing a datebook calendar — but if you take it one step at a time, you've got it made. □

Put Your Best Guess Forward

Most companies go through a budgeting or forecasting session once or twice a year. As with any other prediction, the results are just estimates that become quickly outdated. Typically, the first two months' figures will be fairly predictable, with accuracy dropping off steadily for the following months.

The following Lucid spreadsheet can be used for financial budgeting or planning direct manpower either for an entire department or individual areas within a department. It should make any manager's job easier.

We'll divide the manpower report into the following sections:

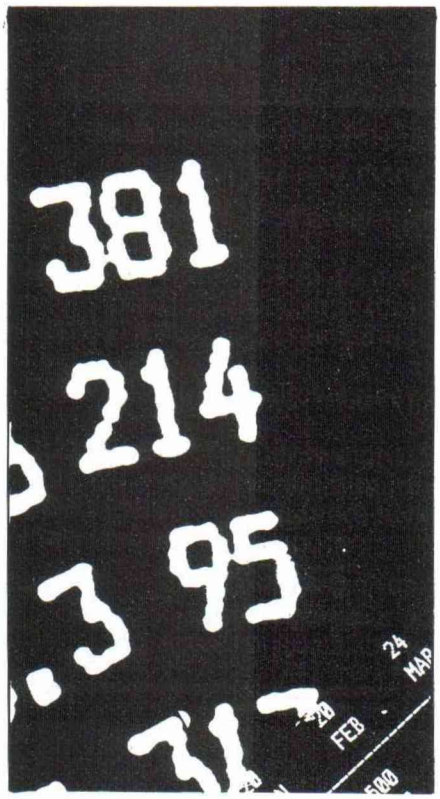
1. Heading
2. Forecast from Marketing
3. Labor Hours
4. Manpower Requirements
5. Financial Data

HEAD IT UP

This spreadsheet is large. It consumes about 16 kilobytes (K) of random access memory (RAM). It uses columns A through AB and 56 rows.

To begin with, set the width of column A to 20 and the widths of columns B through N to eight. The remaining columns should remain at nine.

Type in the file name and the initials of the designer and principle user in



cell A1. Our title will be "FY '86 PRODUCTION — WIDGETS." Use a second line for the name of the preparer of the information or data.

We will want to know when we revise or print the spreadsheet. So in A2 type REVISION:. Move to B2, press *F1* (EDIT) and then Ctrl-D and the date will appear. Change the date whenever you modify or print the spreadsheet. In A4 type DAYS/MONTH. Heading to the right of column A will contain the number of work days in the month and the name of the month. In column B through M, we will input the projected work days for each month.

Our fiscal calendar assumes a 4-4-5 quarter. That means that regardless of the actual calendar, we always have four weeks in the first two months of the quarter and five weeks in the third. Also, our fiscal year begins in July.

The following is a list of months and their respective work days used for the headings in B4 through M4:

JUL	20	JAN	20
AUG	20	FEB	20
SEP	24	MAR	24
OCT	20	APR	20
NOV	20	MAY	20
DEC	18	JUN	23

Move to A5 and type the heading MODEL NUMBER. This represents the information that will appear below in column A.

In N4, calculate the total number of work days with the formula +SUM(B4:M4). Drop the cursor to N5 and type TOTAL.

The production forecast example shows the forecast, in units, for each of five model numbers. This portion of

mstmpr gg/gr — FY '86 PRODUCTION - WIDGETS
REVISION:Jun 25, 1986 — PREPARED BY: GNG

DAYS/MONTH >	20	20	24	20	20	18	20	20	24	20	20	23
MODEL NUMBER	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN
101	1000	1200	1300	1500	1600	1500	1000	1500	1700	1700	2000	1000
201	500	650	700	650	600	600	650	1000	500	500	600	600
301	250	300	300	300	300	300	300	300	300	300	300	400
401	5000	5200	5300	5200	5500	6500	6500	6500	6500	6500	6500	6800
501	1000	1000	1000	1000	900	1100	1100	1100	1100	1200	1300	1200
TOTAL	7750	8350	8600	8650	8900	10000	9550	10400	10100	10200	10700	10000

CURRENT MANPOWER: 18

The monthly production projection section of the spreadsheet.

the spreadsheet will compare our current manpower with our future needs. The units should be typed in as shown, stopping at M11. You'll need to change the default setting so the numbers aren't displayed as decimals.

Move to row 13. In column A type TOTAL. In B12, change the column with the formula +SUM(B7:B11). Replicate this formula from column B to column N.

We'll need a spot to indicate the current number of employees. In A15, type CURRENT MANPOWER. Enter the number in B15 (18 in the example).

Move to cell N7 and total the production quantity forecasted for each model with the formula +SUM(B7:M7). Replicate this to N11.

FY '86			
STD	JUL	AUG	SEP
0.3177	318	381	413
0.3294	165	214	231
0.3177	79	95	95
0.3294	1647	1713	1746
0.3294	329	329	329
==========	==========	==========	==========
	2538	2732	2814

The labor hour calculations section of the spreadsheet.

LABOR HOURS

Each product we produce has a labor standard applied to it. This is the amount of time it will take to build on unit, usually expressed as a decimal. We'll use a separate section of the spreadsheet to calculate total labor hours for each product.

Go to cell O4 and type in the heading FY '86. In O5 type STD. Move the cursor to P5, type +B5, and replicate this across to AB5. With the cursor in S3, type in the title of this section: LABOR HOURS CALCULATIONS.

Move the cursor to cell O7. In O7 and O9, type in .3177. In O8, O10 and O11 type in .3294. Because there are four decimal places, change the default setting for decimals from two to four.

Use the following formulas for column P to calculate labor hours:

Cell	Formula
P7	+B7*$O7
P8	+B8*$O8
P9	+B9*$O9
P10	+B10*$O10
P11	+B11*$O11

Replicate the formulas across to column AA. Note the dollar signs ($). These mean the values in column O will always be referenced and remain absolute and constant.

To total the labor hours associated with each model, move to AB7 and type in the formula +SUM(P7:AA7). Replicate this down to AB11.

Now we'll total the labor hours for each month. In cell P13 type in the formula +SUM(P7:P11) and replicate this across to column AB13.

MANPOWER REQUIREMENTS

This is the section that will do all the work. When completed, all you'll have to do is enter in your forecast, the current number of employees you have, and then press *F2* (Calc). After a few seconds your budget will be done.

Move the cursor to A17 and type MANPOWER REQUIREMENTS. Because each section is designed to be printed individually, we'll repeat some of the information from above. In A19 type DAYS PER MONTH, and in A20 type MONTH. Move the cursor to B19 and type +B4 to bring down the number of days from row 4. We'll do the same thing for the name of the month. Type +B5 in B20. Copy both of these formulas from B4 and B5 across their respective row to column N.

The following entries are for column B, but you should replicate each to column M.

Since we may want to print this section as a single entity, we'll need to have the number of units we will be required to produce each month. This number is available from the labor section of the spreadsheet.

Cell	Entry
A22	TOTAL UNITS
B22	+B13

We'll want to figure out the number of hours required to produce the total units if everyone worked 100 percent of the time, with no absenteeism and no problems.

Cell	Entry
A23	TOTAL HOURS @ 100%
B23	+P13

Because the number of work days differ every month, so will the number of hours one individual will work. We need to account for this.

Cell	Entry
A24	HOURS PER PERIOD
B24	+B19*8

Now we will divide the number of hours at 100 percent by the hours in the period. You can see from our example that we need only 16 people, yet we have 18. Does that mean we have to lay two people off? Not necessarily.

Cell	Entry
A25	TOT MNPWR @ 100%
B25	+B23/B24

Productivity means how effectively the employees work. It takes into consideration inefficiency, employee meetings, training, short duration machine downtime, etc. Some of these things can be controlled by you, the boss. If you wish, you can use your own efficiency factor.

Cell	Entry
A26	PRODUCTIVITY
B26	+.82

This next formula takes our manpower at 100 percent and divides it by the productivity factor. In this example, this tells us we need 19 people to do the work of 16, because the 16 are not 100 percent productive. So instead of laying off two people as we previously thought, it looks like we'll need to hire an additional person.

Cell	Entry
A27	MNPWR @ PRODUCTIVITY
B27	+B25/B26

The next formula is just another way to tell us how inefficient we really are. We will use this information later in our financial section.

Cell	Entry
A28	HOURS @ PRODUCTIVITY
B28	+B23/B26

The next three lines are what I consider uncontrollable items. I assume that four percent of the employees will be absent at any given time. Your rate may be higher or lower.

Cell	Entry
A29	ABSENTEEISM - AVG
B29	+B28*.04

There are times when material arrives that does not meet your specifications. You have to decide whether to return it to the vendor, or rework or

sort it at their cost. If you use your own manpower to rework, you'll need to plan for it. I feel one percent of the hours is sufficient to cover vendor rework. This might be too much for any given month, but over 12 months, it's about right.

Cell	Entry
A30	VENDOR WORK ORDERS
B30	+B28*.01

Sometimes employees have emergencies or illnesses that require taking some vacation time. This makes planning ahead for vacations difficult. I assume one percent straight across the board to cover employee vacations; again, you may need to change this to correspond with your own figures.

Cell	Entry
A31	UNPLANNED VACATIONS
B31	+B28*.01

When I receive a report such as this, I don't want to have to search for information. I want to see the figures near the section I will be needing it for. Also, I believe all manual inputs should be near the top to improve the efficiency of the person inputting the data.

This is simply the current manpower times the number of hours in the period that project our manpower needs.

> Employees are not 100 percent efficient.

Cell	Entry
A32	CURRENT MANPOWER
B32	+$B15

The next formula takes the current manpower times the number of hours in the period that project our manpower needs.

Cell	Entry
A33	CURRENT HOURS
B33	+B32*B24

Enter some blank lines to offset the above from the calculations below.

Cell	Entry
A34	*blank*
B34	*blank*

DAYS PER MONTH >	20	20	24
MONTH	JUL	AUG	SEP
TOTAL UNITS	7750	8350	8600
TOTAL HOURS @ 100%	2538	2733	2814
HOURS PER PERIOD	160	160	192
TOT MNPWR @ 100%	16	17	15
PRODUCTIVITY	0.82	0.82	0.82
MNPWR @ PRODUCTIVITY	19	21	18
HOURS @ PRODUCTIVITY	3095	3333	3432
ABSENTEEISM - AVG	124	133	137
VENDOR WORK ORDERS	31	33	34
UNPLANNED VACATION	31	33	34
CURRENT MANPOWER	18	18	18
CURRENT HOURS	2880	2880	3456
PROPOSED HOURS	3281	3533	3638
PROPOSED MANPOWER	21	22	19
**********			
MNPWR VAR/ACTUAL	-3	-4	-1
O.T. REQ'D @ EQUIV	401	652	181
**********			
HOURS TO LEVEL	0	-601	0

The manpower requirements section of the spreadsheet.

Now take the hours (incorporating our productivity factor) and add the uncontrolled hours of absenteeism, unplanned vacations and vendor rework.

Cell	Entry
A35	PROPOSED HOURS
B35	+SUM(B28:B31)

Now divide the proposed hours by the hours in the period.

Cell	Entry
A36	PROPOSED MANPOWER
B36	+B35/B24

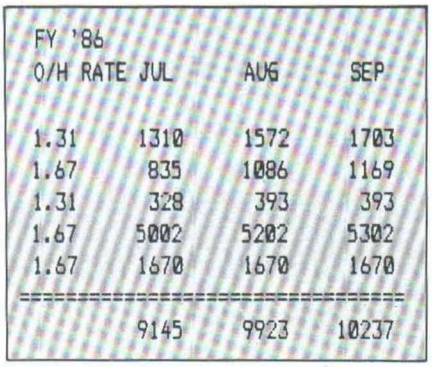
Here, we'll subtract the proposed manpower from the current manpower. If the number is negative, we'll need to hire some employees; if it's positive, we'll transfer or lay off some people.

Cell	Entry
A38	MNPWR VAR/ACTUAL
B38	+B32-B36

If the current hours are greater than the proposed hours, obviously we won't need to work any overtime, so the next formula will print out a zero. If overtime is required the formula will calculate how many hours and print it out. This should aid you in your decision to either hire more people or work more hours, whichever is more economically feasible. In our example, this report was run on June 25th, and it shows a need to work 399 overtime hours in July or hire two people. Hiring two people and training them takes time, but since August shows the same need it might be wise to hire them and work overtime until they reach an efficient pace.

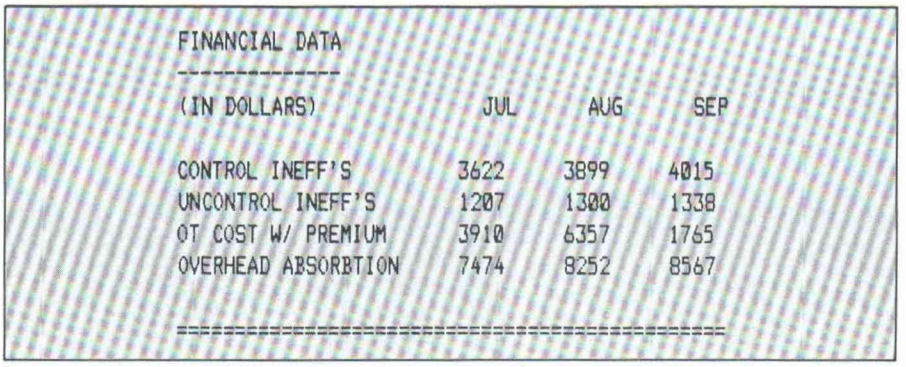
Cell	Entry
A39	O.T. REQ'D @ EQUIV
B39	+(B35>B33)&(B35=B33) +B35-B33

Notice we did not place a formula in B42 because we are comparing one month with the previous one. The formula is saying that if the manpower in August is less than July, then subtract July from August and multiply the result by the hours in the month. If this is not true, then print a zero. Our example says we need 601 hours to level September with August. If we look



FY '86			
O/H RATE	JUL	AUG	SEP
1.31	1310	1572	1703
1.67	835	1086	1169
1.31	328	393	393
1.67	5002	5202	5302
1.67	1670	1670	1670
	9145	9923	10237

The overhead absorption section of the spreadsheet.



FINANCIAL DATA			
(IN DOLLARS)	JUL	AUG	SEP
CONTROL INEFF'S	3622	3899	4015
UNCONTROL INEFF'S	1207	1300	1338
OT COST W/ PREMIUM	3910	6357	1765
OVERHEAD ABSORBTION	7474	8252	8567

The financial data section of the spreadsheet.

ahead to October, November December, you'll see we have a bubble in December and a small problem in both October and November. To level the manpower in September we can either move some of the load from August into September and move some hours from October and possibly November.

You now have enough information to discuss the problem intelligently. Bear in mind the declining accuracy of the forecast: Our example starts with June, which means July and August are somewhat accurate, September probably isn't.

Cell	Entry
A42	HOURS TO LEVEL
B42	*blank*
C42	+(C36<B36)&(C36=B36) +((C36-B36)*C24)

The last thing to do with this section is to total some of the rows. Rows 25, 27, 32 and 36 are not totalled. Row 26 is averaged. The reason is that most companies would not have a constant productivity factor. The formula to average is:

+SUM((B26:M26)/CNT(B26:M26)

The remaining rows are totalled with the formula +SUM(B22:M22), with the row number changing respectively.

THE FINANCIAL DATA

Because so much information appears in this report, we have to do some further calculations for our budget. Go to A46 and type FINANCIAL DATA. In A48 type (IN DOLLARS). In columns B48 to N48, replicate the formula +B5 to bring down the month headings.

Again, the following formulas are for column B, these should be replicated to column M.

Our next formula simply takes the difference between the hours at 100 percent and the hours with our productivity factor and multiplies the result by our average wage rate, in this case $6.50.

Cell	Entry
A50	CONTROL INEFF'S
B50	+((B28-B23)*6.50

The next formula is similar to the one above, except we are costing our uncontrollable inefficiencies such as absenteeism, vendor rework and unplanned vacations.

Cell	Entry
A51	UNCONTROL INEFF'S
B51	+SUN(B29:B31)*6.50

Because overtime is usually one and one-half times the normal rate of pay, we multiply the overtime hours times the average overtime rate.

Cell	Entry
A52	OT COST W/PREMIUM
B52	+B39*(1.5*6.50)

Every product that we produce contributes to our overhead, so some of the money we earn from the sale of our product has to pay for the wages of our indirect labor. Each product contributes a certain percent of its value.

Cell	Entry
A53	OVERHEAD ABSORPTION
B53	+P28

We need to design a section to perform our overhead calculations. Move the cursor to O19 and type in FY '86, and O/H RATE in O20. In columns P20 to N20, type the names of the months. At S18, type OVERHEAD ABSORPTION CALCULATIONS.

Below are the entries for columns O and P; the formulas in column P should be replicated across to column M.

Cell	Entry
O22	1.31
O23	1.67
O24	1.31
O25	1.67
O26	1.67
P22	+B7*$O22
P23	+B8*$O23
P24	+B9*$O24
P25	+B10*$O25
P26	+B11*$O26
P27	"=========
P28	+SUM(P22:P26)

We'll want to know the total overhead for each of our models, so move the cursor to N22. The formula is +SUM(P22:M22); this should be replicated to M26. In M28, find the total with +SUM(N22:N26). Then return to N50 and total our financial data with the formula +SUM(B50:M50), replicated to N53.

If you haven't already, set the default decimal places to zero for the entire spreadsheet, with the exception of the row 26, column O, rows 7 through 11 and rows 22 through 26.

CONCLUSION

This particular spreadsheet deals strictly with direct labor, but you could carry it a step further by adding lines for indirect labor. Another option would be to cost the entire labor hours by adding a line in the financial section. It would be just as easy to add a cost of sales for each month. If your department is into zero base budgeting, you have the justification for your department right here. □

TEXTMERGE *By Alan L. Zeichick*

Do-It-Yourself Lette

Use TextMerge to create personalized letters from your mailing list.

Quick: What's the first program that's usually implemented on any new computer?

If you answered BASIC, you're probably right. But what about the first application? Well, I'd guess text formatting.

Even though the Model 100-series of computers has a built-in text editor, the early issues of *Portable 100* are filled with numerous ads for third-party, cassette-based text formatting software.

The earliest such software was written in BASIC; later, adventurous programmers developed assembly-language products. Today most high-powered laptop text formatters are in ROM — such as Super ROM and Ultimate ROM II — but BASIC text formatters are still available.

TEXT.BA, which works on the Model 100 and Tandy 102 and 200, is written entirely in BASIC, with no nasty peeks, pokes or calls. Plus it provides a powerful mail-merge feature (as pioneered by microcomputer text formatters like Wordstar). And the whole thing fits into 2.4K of RAM.

TEXT.BA started out life as a simple mail-merge program called MERGE.BA. But after the merge program was written, it seemed silly not to turn it into a regular text formatter. That's why the program has some interesting limitations. To run TEXT.BA as a text formatter, you'll need a document file. But the file can't be too big — say, no larger than 5,000 characters — unless you want to change the size of the CLEAR statement. Why? Because the program copies the entire contents of the document into an array (DA$, for document array), instead of reading the file one line at a time like most text formatters.

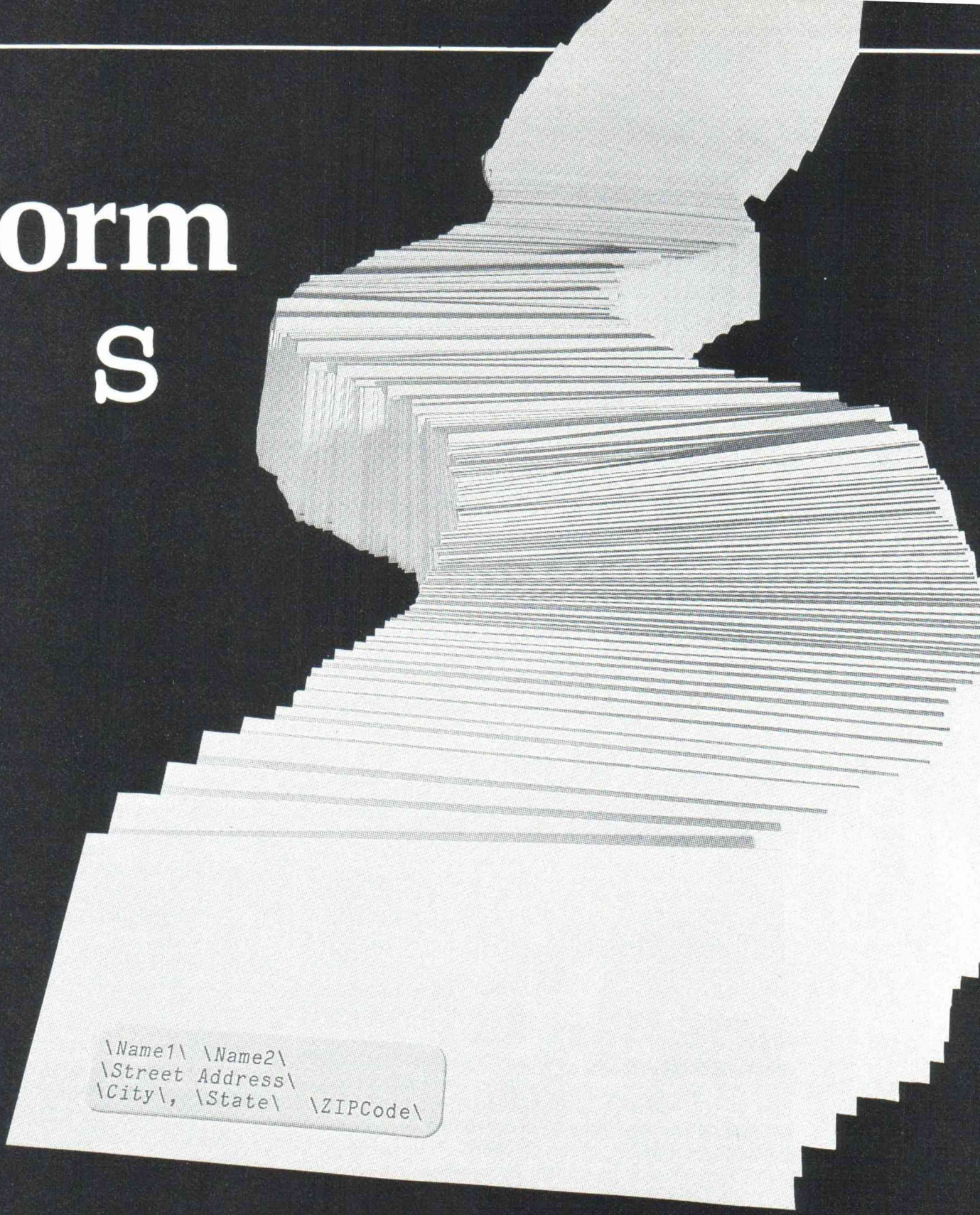
From the main menu, select TEXT.BA. The program will prompt you for the name of a document file; enter the name here. The program then pauses for a few seconds to read the document. If the file's not found or if it exceeds the amount of available string space, you'll see an appropriate error message.

You'll then be asked for the name of a merge file. That's the one that contains the data to be used in making form letters. We'll get to that shortly; for now, press ENTER again to indicate that you're just formatting a single document. TEXT.BA now prints the document on the attached parallel printer. (It's not a very difficult modification to use a serial printer. Just change the LPRINT statements to PRINT #3 and open the serial port for output as device number three.)

Your printed output will be formatted according to parameters given at the start of the program. You have full control over the size of the left-hand

PHOTOGRAPH BY BENJAMIN MAGRO

Form
r s



margin, the maximum length of a printed line, the total size of the paper and the number of lines that actually print on a page. These features are handy not only for calibrating your printer, but also for printing on odd-size forms, such as custom invoices or continuous-form postcards.

TEXT.BA gives you control over several other print features, too. If you're using single-cut sheets of letterhead, or if your printer doesn't have a tractor feed, you can instruct TEXT to stop after each page, and wait for you to load another sheet. You can select single or double spacing. Finally, you can tell TEXT to send a line feed after each carriage return. Some early Tandy printers require an explicit line feed character (ASCII 10), whereas most others assume a line feed after a carriage return (ASCII 13). The basic rule: If your printer prints line after line without advancing the paper, you need to select line-feed mode. If the printer double-spaces when you've actually selected single-spaced text, you don't want line-feed mode.

That's the simple application of TEXT.BA. But more is in store: mail merge.

FORMS UNLIMITED

The expression *mail merge* harks back to the original use of this process; taking names from a list and generating individual letters. You're all familiar with them, since they often appear as advertising promotions for sweepstakes. The process works by taking a sample document called the *template*, and a database called the *merge file*. The template letter has special fields, highlighted by *delimiters*. In our case, the delimiter is the backslash character, typed as Grph-Minus on the laptop keyboard.

Within the template, regular text is mixed with *field names* surrounded by backslashes. One example might be *Dear \Name\;* another might be *Your payment of \Amount\ is overdue.* In those examples, the actual name or amount will be substituted when a form letter is printed — and will vary for each letter.

The merge file is a little bit more complex. It's actually a database — and it's possible to write a BASIC program that creates and modifies this database.

The first line of the merge file contains the number of distinct fields.

The second line contains all the field names, separated by commas, with no spaces after the commas. These names must be identical to those used by the document template. You don't have to use all fields in your document — but every name in the document must match a field in the merge file. An example of the second line would be *Name,Address,City,State,ZIP.*

The following lines contain the actual data. In this example, each line must have exactly five fields; if a field is blank, simply put in two commas.

If a data item contains a comma, the item must be surrounded by double quotes. Were I a medical doctor, my line field would read *"Alan Zeichick, MD", etc.* Contrary to the rules of English punctuation, remember that the delimiting comma goes outside the closing quote.

Those are the basics of setting up a merge file; once the file's created, be

sure you've got the right number of items on each data line. Then make up your document using the exact field names defined in the merge file. If you use a field name in your template that you've not defined in your merge file, or you make a typographical error, TEXT.BA will give you an appropriate error message:*\Naem\ isn't in the merge file.*

> Today most high-powered laptop text formatters are in ROM.

One error that TEXT.BA *won't* warn you about is failing to place the closing backslash on a variable name while you're editing the document. So, if you get an unusual error message, that should be the first thing to check.

When TEXT.BA displays an error message, it'll give you two options. Pressing Esc returns you to the main menu; that's the option to pick if you have an erroneous field name. Pressing any other key re-runs TEXT.BA from the start; do that when you type in he template or merge-file name incorrectly. If TEXT.BA advises you that you need to CLEAR more string space or redimension an array, press Shift-Break and modify the appropriate program lines.

In case you're wondering what to do with mail-merge, here are a few starter ideas. I'm sure you'll come up with several of your own once you get started:

- Create mailing labels by setting the page length (LP$) equal to six and by placing the appropriate fields into your document template.
- Produce custom invoices by adjusting the page length and margins to the specifics of your pre-printed forms, or by creating the form as part of your template. This applies to computer generated checks too.
- Use TEXT.BA to print library cards.

Note that if you're using a disk drive (such as the Tandy DVI or portable disk drive), and you have a random access operating system that lets you use disk files for input and output, you

```
1000 ' TextMerge
1010 ' Alan L. Zeichick, Portable 100
1020 '
1030 ' ============================
1040 ' Main Routine
1050 '
1060 ' Initialize, display banner
1070 '
1080 CLEAR 5000
1090 MAXFILES=3
1100 ON ERROR GOTO 3430
1110 DIM FA$(25), DA$(50)
1120 DEFINT A-Z
1130 LM=10
1140 PW=60
1150 LP=66
1160 PP=55
1170 PS$="Y"
1180 LF$="Y"
1190 DS$="N"
1200 BS$="\"
1210 RP$=CHR$(27)+"p"
1220 NP$=CHR$(27)+"q"
1230 CLS
1240 LINE (6, 55)-(233, 55)
1250 PRINT @231, RP$; SPACE$(5); "TextMerge by Alan L. Zeichick"
     SPACE$(4); NP$;
1260 IF LF$="Y" THEN LF$=CHR$(10) ELSE LF$=""
1270 '
1280 ' Read letter in da$
1290 '
1300 GOSUB 1730
1310 '
1320 ' Read merge file into fa$
1330 '
1340 GOSUB 1490
1350 '
1360 ' Compare letter to merge file
1370 '
1380 GOSUB 2140
1390 '
1400 ' Read merge file, produce letters
1410 '
1420 GOSUB 2290
1430 '
1440 MENU
1450 '
1460 ' ============================
1470 ' Put merge file names into fa$
1480 '
1490 PRINT @0, "What's the name of the merge file?"
1500 NA$=""
1510 INPUT "Press Enter for no merging"; NA$
1520 PRINT @0, SPACE$(120)
1530 IF NA$="" THEN RETURN
1540 IF INSTR(NA$, ".")=0 THEN NA$=NA$+".DO"
1550 OPEN NA$ FOR INPUT AS #1
1560 '
1570 ' Initialize variables, read from file
1580 '
1590 INPUT #1, FI
1600 IF FI=0 THEN ERROR 50
1610 '
1620 ' Read individual item names
1630 '
1640 FOR FC=0 TO FI-1
1650   INPUT #1, FA$(FC)
1660   NEXT FC
1670 '
1680 RETURN
1690 '
1700 ' ============================
1710 ' Read the letter into da$
```

can access those files with TEXT.BA.

NO ERRORS HERE

I've tried to write TEXT.BA in a reasonably coherent and structured fashion, but it's difficult to write true structured programs with the Model 100 BASIC. So I originally developed TEXT.BA on my Tandy MS-DOS computer, using the Microsoft GW-BASIC's WHILE statement, then rewrote the loops using IF-GOTO combinations on the Model 100. The GOTO statement is only used for make and exit loops.

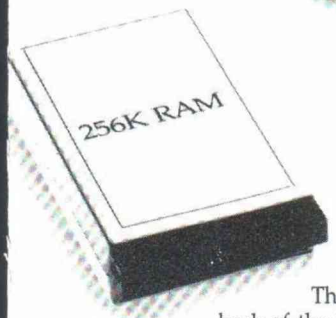
Actually there's one other use of the GOTO: error-trapping statements. There are many reasons why the program can crash: a file is missing or its name is incorrectly given, running out of data, duplicating a file name or a field name is missing from the merge file. Some error conditions are noticed by BASIC, like running out of string space; others are software related, such as specifying an invalid field name.

There's no reason to treat BASIC errors and software errors differently. With its ON ERROR GOTO and

```
1720 '
1730 PRINT @0,
1740 NA$=""
1750 INPUT "What's the name of the document"; NA$
1760 IF NA$ = "" THEN MENU
1770 PRINT @0, SPACE$(80);
1780 IF INSTR(NA$, ".")=0 THEN NA$=NA$+".DO"
1790 OPEN NA$ FOR INPUT AS #2
1800 PRINT @281, RP$; RIGHT$(SPACE$(30)+"Loading document ", 38); NP$;
1810 '
1820 ' Initialize variables, read from the file
1830 '
1840 DR$=""
1850 DI=0
1860 FOR DR=1 TO 100
1870    IF EOF(2) THEN 1940
1880    DR$=DR$+INPUT$(1, 2)
1890    NEXT DR
1900 '
1910 ' Keep reading while there's data; if there's a
1920 ' backslash, isolate the field with BS$
1930 '
1940 IF LEN(DR$)<= 0 THEN RETURN
1950    DP=INSTR(DR$, BS$)
1960    IF DP=0 THEN DA$(DI)=DR$ ELSE DA$(DI)=LEFT$(DR$, DP-1)
1970    DI=DI+1
1980    GOSUB 2050
1990    IF DP>0 THEN DP=INSTR(DR$, BS$): DA$(DI)=BS$+LEFT$(DR$, DP-1)
        :DI=DI+1:GOSUB 2050
2000    GOTO 1940
2010 '
2020 ' ------------------------
2030 ' Subroutine to fill dr$
```

256K RAM FOR THE M102 & M100

Memory doesn't erase when you unplug it —
Buy multiple cartridges and carry them with you anywhere!



Instantly upgrade your Model 102 or your old Model 100 to 256K of RAM or use the budget unit for 160K. That little cartridge simply plugs onto the back of the Model 102. That's all there is to it. Buy several if you like, because they don't lose your files when you unplug them. (For the older type Model 100 we have a special shape that goes inside the little back compartment).

Like eight Model 102's in one.

You then have 8 banks of RAM of 32K each. The additional seven banks also work just like your Main Menu. Push a function key: you are in the second bank. Push again, the third and so forth until you are back to your original bank. Its built-in NiCad battery recharges right from the Model 102 and is guaranteed for a full year.

Copy a file from bank to bank with a function key.

Each bank is like having another Model 102. All the built-in programs and snap-in ROMs appear and work in all banks. Your widebar cursor moves from file to file and you just press ENTER.

Data files in excess of 220K.

You aren't limited to working with banks of 32K each. The ROM software included has extensions to BASIC that allow modification of programs to write and read data throughout the entire 256K. Data files can be created using all available memory, appearing to the user as one continuous 256K of RAM.

More features.

Rename or kill any file. Set the date, day and time with function key ease. Even use non-Radio Shack printers without those tricky dipswitch settings. Whenever you take out the snap-in ROM it leaves a tiny program that lets you switch banks just by pressing ENTER.

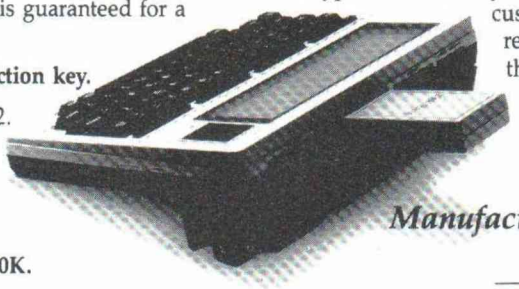
Total RAM*	128K	160K	256K
Model 100	$299	$369	$549
Model 102	—	$369	$549

**Including original 32K*

The complete system.

The small center photo is the new 256K cartridge with the PCSG/Cryptronics 6 ROM bank, that allows six software ROMs installed, plus gives 30 hours of battery power to your computer. Combine that with SuperROM and the Holmes/PCSG portable disk drive (350K on a diskette): the complete portable system.

It is truly an engineering masterpiece. Thousands of PCSG/Cryptronics memory expansions have been shipped to satisfied customers — more are installed and working reliably than any competitive product. We sell these memory expansions on a 30-day trial. Simply return it within 30 days for a full refund if you are not satisfied.



Manufactured by Cryptronics, Inc.

PERSONAL COMPUTER SUPPORT GROUP
11035 Harry Hines Blvd. #207 • Dallas, Texas 75229
214-351-0564

MasterCard VISA

ERROR statements, and the ERR (return error code) and ERL (return error line), we can handle all errors in one place, at the same time.

The beginning of TEXT.BA contains the statement ON ERROR GOTO 3430. Any error condition, therefore, will cause a branch to that line.

When an error condition occurs, either naturally or through the use of the ERROR statement, the computer enters an "error mode" and branches to the line specified by the ON ERROR GOTO. While in error mode, the built in function ERR can be used to test which error occurred, and the function ERL the line that triggered the error condition. The error numbers, by the way, are listed in an appendix in the back of your laptop's manual; the same numbers are used for the Model 100, Tandy 102 and Tandy 200.

Our error-handling routine tests for several machine-generated error conditions and prints a more verbose response than the normal cryptic error message. Error number four is a syntax error (?SN); chances are the program was typed incorrectly. Error nine's mnemonic, ?BS, means Bad Subscript; one of the arrays exceeded its bounds and should be enlarged. Error 14 (?OS) says that BASIC is out of string space; this means the document template was larger than normal, and the CLEAR statement in line 1080 should be changed.

If the printer isn't connected or operational, error 18 will occur. If one of the requested file names is missing, error 52 will result. If a file name is illegal (has unusual embedded characters), error 55 will occur. And if you specify the same file name for both the merge and template, BASIC normally responds with an ?AO error, number 53.

In addition to those errors, TEXT.BA uses the ERROR statement to simulate new error codes. I've used those that BASIC ignores: If the ON ERROR GOTO 3430 statement is removed, these error codes will be displayed as ?UE, or undefined error. If the merge file doesn't have a number greater than zero as the first line, TEXT.BA signals error 50. If a field name's missing, that's error 59. Finally, if the program runs out of data while reading from the merge file, that's okay: That's error 54, and signals a return to the main menu.

The error message is assigned to the string variable MS$ (for *message*), and the error handler terminates with the RESUME statement. RESUME must be used to terminate an error handler;

```
2040 '
2050 IF DP=0 THEN DR$="" ELSE DR$=MID$(DR$, DP+1)
2060 FOR DR=1 TO 100-LEN(DR$)
2070   IF NOT EOF(2) THEN DR$=DR$+INPUT$(1, 2)
2080   NEXT DR
2090 RETURN
2100 '
2110 ' ============================
2120 ' Compare da$'s fields to fa$
2130 '
2140 IF FI=0 THEN RETURN
2150 FOR DC=0 TO DI
2160   IF LEFT$(DA$(DC), 1)<>BS$ THEN 2210
2170   FOR FC=0 TO FI
2180   IF MID$(DA$(DC), 2)=FA$(FC) THEN DA$(DC)=BS$+STR$(FC) :GOTO 2210
2190     NEXT FC
2200   ERROR 59
2210   NEXT DC
2220   RETURN
2230 '
2240 ' ============================
2250 ' Read records from FIL, merge with DOC
2260 '
2270 ' Load the first name into fa$
2280 '
2290 DD=0
2300 GOSUB 2370
2310 GOSUB 2490
2320 GOTO 2300
2330 '
2340 ' ============================
2350 ' Read from FIL into fa$
2360 '
2370 IF FI=-1 THEN MENU ELSE IF FI=0 THEN FI=-1:RETURN
2380 FOR FC=0 TO FI-1
2390   INPUT #1, FA$(FC)
2400   NEXT FC
2410 RETURN
2420 '
2430 ' ============================
2440 ' Merge and print a form letter
2450 '
2460 ' Increment the letter counter, print banner
2470 ' Remember to pause between pages if necessary
2480 '
2490 DD=DD+1
2500 LC=0
2510 '
2520 PRINT @281, RP$;
2530 IF PS$="Y" THEN PRINT "     Load paper, then press a key     ";
     :IF INPUT$(1)=CHR$(27) THEN MENU
2540 PRINT @281, RIGHT$(SPACE$(30)+"Printing letter"+STR$(DD)+" ",
     38); NP$;
2550 '
2560 ' Load up the buffer with the first record; remember
2570 ' to substitute from fa if it begins with BS$
2580 ' Also remember to print the left margin
2590 '
2600 LPRINT SPACE$(LM);
2610 CC=0
2620 DC=0
2630 DG=PW
2640 BF$=""
2650 IF LEFT$(DA$(0), 1)=BS$ THEN GOSUB 3350 ELSE BF$=DA$(0)
2660 DC=1
2670 '
2680 ' Keep on looping until it runs out of document
2690 ' If the buffer's nearly empty, refill it
2700 ' dg is the characters needed to finish the line
2710 '
2720 IF LEN(BF$)<DG+20 THEN IF LEFT$(DA$(DC), 1)=BS$ THEN GOSUB 3350
     :DC=DC+1 ELSE BF$=BF$+DA$(DC):DC=DC+1
```

once the error occurs, any future error trapping is suspended until the RESUME is executed.

After the RESUME statement sends the message to line 3560, and the message is printed, you're given the choice of returning to the menu, rerunning TEXT.BA, or pressing Shift-Break to jump to BASIC. Isn't that friendlier than a *?FF Error in 2300?*

BACK TO THE STRUCTURE

TEXT.BA is broken down into several distinct parts, each of which is a complete subroutine:

Lines 1080 through 1440 initialize all variables, display the opening banner and call various subroutines.

Lines 1490 through 1680 open the merge file, read the number of merge field into the variable FI, then read the field names into the string array FA$. If no name was given for the merge file, this routine is skipped.

Lines 1730 through 2000 open the template file and read the contents of the file into the string array DA$. Each element of the array may contain as many as 100 characters of the template, taken from a buffer, DR$. If the subroutine encounters a replacement field (surrounded by backslashes), that field name is given its own DA$ element, prefaced by the backslash character.

Lines 2050 through 2090 help the previous routine by filling the DR$ buffer from the template file.

Lines 2140 through 2220 compare the replacement fields found in the template with the merge file's field names.

Today most high-powered laptop text formatters are in ROM.

If a match is found, the template's field name is replaced by the FA$ element number for quick access. If a template field isn't matched by an entry in the merge file, error 59 results.

Lines 2290 through 2320 call two other subroutines to read data from the merge file and to produce the form letter. A GOTO statement turns those two subroutine calls into an infinite loop.

Lines 2370 through 2410 read from the merge file into the merge array FA$. That's the same array previously used for the field names; the names aren't needed any longer, so re-using the array helps conserve precious string space and RAM.

Lines 2490 through 3020 are the meat of the program, comprising the main text-formatting loop. The loop's internal comments help explain this loop,

```
2730 '
2740 ' Set bp to the location of any carriage return
2750 '
2760 BP=INSTR(LEFT$(BF$, DG), CHR$(13))
2770 '
2780 ' If carriage return, print up to it and then a new line
2790 '
2800 IF BP>0 THEN LPRINT LEFT$(BF$, BP-1);:BF$=MID$(BF$, BP+2)
     :GOSUB 3070
2810 '
2820 ' If no carriage return and buffer's long, find word break
2830 '
2840 IF BP=0 AND LEN(BF$)>DG THEN FOR DG=DG TO 1 STEP -1:IF MID$(BF$,
     DG, 1)=" " THEN 2890 ELSE NEXT DG
2850 '
2860 ' If no carriage return and buffer's short, print it all;
2870 ' If the buffer's long, print some of it and then a new line
2880 '
2890 IF BP=0 THEN IF LEN(BF$)<DG THEN LPRINT BF$;:CC=CC+LEN(BF$):
     BF$="" ELSE LPRINT LEFT$(BF$, DG);:BF$=MID$(BF$, DG+1)
     :GOSUB 3070
2900 '
2910 ' Determine how many characters are needed, continue
2920 '
2930 DG=PW-CC
2940 IF LEN(BF$)>0 OR DC<DI THEN 2720
2950 '
2960 ' Print blank lines to finish off the page
2970 '
2980 FOR LC=LC TO LP-1
2990   LPRINT LF$
3000   NEXT LC
3010 '
3020 RETURN
3030 '
3040 ' Subroutine for a new line
3050 ' ------------------------
3060 '
3070 LPRINT LF$
3080 IF DS$="N" THEN LC=LC+1 ELSE LPRINT LF$:LC=LC+2
3090 CC=0
3100 '
3110 IF LC<PP THEN LPRINT SPACE$(LM);:RETURN
3120 '
3130 ' If necessary, finish the page
3140 ' Print the blank lines needed to end the page
3150 '
3160 FOR LC=LC TO LP-1
3170   LPRINT LF$
3180   NEXT LC
3190 '
3200 ' If pause between pages, then prompt the user
3210 '
3220 LC=0
3230 PRINT @281, RP$
3240 IF PS$="Y" THEN PRINT "     Load paper, then press a key     ";
     :IF INPUT$(1)=CHR$(27) THEN MENU
3250 PRINT @281, RIGHT$(SPACE$(30)+"Printing letter"+STR$(DD)+" ",
     38); NP$;
3260 '
3270 ' Print the left margin and return to the letter
3280 '
3290 LPRINT SPACE$(LM);
3300 RETURN
3310 '
3320 ' ------------------------
3330 ' Subroutine for a replacement field
3340 '
```

continues on page 58

YOU BE THE EDITOR

Every reader has unique needs. Some seek guidance from reviews of software and peripherals. Others look to a magazine to provide tips to increase their programming proficiency. Accounts of innovative applications attract a big audience, as do articles about other users who find creative ways to use their systems.

You can help tailor **Portable 100** to your unique needs by rating the articles that appeared in this issue. Just circle the numbers indicated below on the Reader Service Card at left. Rate all the articles, some of them or just one. Then detach the card and drop it into the mail — we pay the postage.

The editors will analyze your feedback and respond by commissioning more articles of the most popular types — and fewer of the losers.

HEIR TO THE THRONE
The Tandy 102.
(page 6)
Excellent 160
Good 161
Poor 162

WEIGHING THE DIFFERENCES
Model 100 vs. Tandy 102.
(page 34)
Excellent 163
Good 164
Poor 165

BUILD THE DATAMITE
Low-cost ADC.
(page 20)
Excellent 166
Good 167
Poor 168

HERE COMES THE SUN
Track it.
(page 26)
Excellent 169
Good 170
Poor 171

INDUSTRIAL STRENGTH MEMORY Bubble Memory.
(page 30)
Excellent 172
Good 173
Poor 174

ROM IT YOURSELF
Burn your own.
(page 32)
Excellent 175
Good 176
Poor 177

PROGRAMMING WORKSHOP
Step-by-step.
(page 38)
Excellent 178
Good 179
Poor 180

PUT YOUR BEST GUESS FORWARD Make accurate predictions. (page 44)
Excellent 181
Good 182
Poor 183

DO-IT-YOURSELF FORM LETTERS Use TextMerge.
(page 48)
Excellent 184
Good 185
Poor 186

OVERALL QUALITY OF THIS ISSUE

Excellent 187
Good 188
Poor 189

ADVERTISERS INDEX

Free Information

For free information on products advertised in this issue of Portable 100, locate the Reader Service number corresponding to the advertisement that interests you. Circle the number on the card at left and drop it into the mail. The literature you've requested will be forwarded to you without obligation. Please allow 3-5 weeks for delivery.

SAVE 48% RIGHT NOW!

"I have found something in almost every issue that has been worth the price of the subscription. I am amazed at the broad spectrum of the technical articles you publish, with ample material for both novice and advanced programmer."

Roger L. *Holstege*. . . MD

And with 6 to 12 new programs for your Tandy computer in every issue, you too will find something each month that's worth the low cost of your subscription! Just type them in, and they're yours. Spreadsheets, word processors, all kinds of home, business and personal applications, plus helpful **utilities** that make all your programs easier to write, debug and use.

Valuable Hardware Projects

Want more value? How about *80 Micro*'s money saving **hardware projects** you build yourself. . . or our frequent **patches and enhancements** that make commercial software programs even better.

Programming Techniques & Tutorials

You'll add to the value of your computer and get more out of every hour you spend computing with expert guidance from *80 Micro*'s **programming techniques** that help you streamline and improve programs. In addition, *80 Micro*'s step-by-step **tutorials** keep you ahead of today's fast-moving technologies like nothing else.

Hardware & Software Reviews

Avoid even one chancey hardware purchase, and you've saved enough to repay years of *80 Micro* subscriptions! You'll save yourself a bunch of time and hassle, too, with *80 Micro* checking out new software for you. Zero in on programs and products you want and skip the clunkers.

► For even quicker service, ☎ **CALL TOLL FREE 1-800-258-5473** (in NH, dial 1-924-9471) and charge it to your credit card!

The Most Widely Read Magazine In Its Field

80 Micro is the most widely read magazine serving Tandy users today, with the largest editorial staff in its field. It's the oldest and most authoritative, too, so you know you can trust the help and advice you find in every issue.

Start Saving Now With 48% Off The Cover Price!

Yes it's true. You don't even have to wait for your first issue to start saving with *80 Micro*! Just use the coupon below or return the postpaid card opposite and you'll start receiving *80 Micro* every month at a full **48% off** the Newsstand Rate. ***If you're ever dissatisfied, for any reason, you'll be reimbursed for all un-delivered issues. Guaranteed!*** So do it now. And get *80 Micro*'s value working for you, *every* month from now on!

* TRS-80 is a trademark of Radio Shack, a division of Tandy Corp.

YES, I want to add the value of *80 Micro* every issue. Enter my no-risk subscription for 12 monthly issues at the Introductory Rate of just $24.97! I save a full **48% off** the newsstand price!

☐ Payment enclosed ☐ Bill me

Name (Please Print)

Address

City State Zip

Canada and Mexico, $27.97, Foreign surface $44.97, 1 year only. Foreign airmail, $79.97. US funds drawn on US bank. Please allow 6–8 weeks for delivery.

378B8

80 Micro • Box 981 • Farmingdale, NY 11737

on a line-for-line basis.

Lines 3070 through 3300 are executed whenever the text-formatting loops call for a new printed line. The subroutine prints a carriage return and optional linefeed, followed by the new left margin. It then tests to see if a new page is necessary; if so, it prints enough blanks to finish the page.

Lines 3350 through 3380 take care of substituting a replacement field for actual template text. Here's how it works: In line 2650 or 2720, the first character of the current element of the template buffer, DA$(DC), is compared to a backslash. If it is, then the routine at 3350 knows that the element contains the number of the appropriate element in the merge array, FA$. The number is extracted: MID$(DA$(DC), 2)). This expression is used as the appropriate subscript to FA$.

Finally, lines 3430 through 3620 handle all of the error conditions as described earlier.

When you go to type in TEXT.BA, do yourself a favor: Don't type the comment and blank lines. No GOTO statements reference comments, so you're safe. You can also compress the program into multiple-line statements where appropriate — but be wary of the targets of GOTO statements.

If you'd like to modify TEXT.BA, here are some suggestions:

- Add an optional setup string, SS$ which will be printed at the top of each page. This can change the printer pitch, character set and other defaults.

```
3350 HD$=FA$(VAL(MID$(DA$(DC), 2)))
3360 IF LEN(HD$)>1 AND LEFT$(HD$, 1)=" " THEN HD$=MID$(HD$, 2)
3370 BF$=BF$+HD$
3380 RETURN
3390 '
3400 ' ============================
3410 ' Error handling
3420 '
3430 MS$ = "Error number"+STR$(ERR)+" in line"+STR$(ERL)
3440 IF ERR=4  THEN MS$="Syntax error in line "+STR$(ERL)
3450 IF ERR=9  THEN MS$="You need to re-DIM your arrays."
3460 IF ERR=14 THEN MS$="You need to CLEAR more string space."
3470 IF ERR=18 THEN MS$="Input/Output error in line"+STR$(ERL)
3480 IF ERR=50 THEN MS$=NA$+" isn't a merge file."
3490 IF ERR=52 THEN MS$="Can't find the file, "+NA$+"."
3500 IF ERR=53 THEN MS$="You can't use "+NA$+" twice."
3510 IF ERR=54 THEN MENU
3520 IF ERR=55 THEN MS$=NA$+" is an illegal file name."
3530 IF ERR=59 THEN MS$=BS$+MID$(DA$(DC), 2)+BS$+" isn't in the
     merge file."
3540 RESUME 3560
3550 '
3560 ON ERROR GOTO 0
3570 CLS
3580 PRINT MS$
3590 PRINT
3600 PRINT "Press any key to re-run, Esc for Menu."
3610 IF INPUT$(1)=CHR$(27) THEN MENU ELSE RUN
3620 END
```

- Make up some default variable names, like DATE or TIME, which will always be the current date in English (like September 1, 1986).

- Use a variable PN$ to keep track of page numbers. If PN$ is set equal to "N" at the start of the program, no page numbers will be printed. If PN$ is "Y," print page numbers at the bottom of each page.

- Have TEXT.BA read the setup information from a file, which will make the data easier to examine and modify.

- Teach the program to print multiple copies of a given document as determined by a variable MC. Hint: Place a FOR-NEXT loop around the GOSUB statement at line 2310.

- If you're a more intrepid programmer, create a main menu-like screen for selecting the document and merge-file names. □

What gives you rapid access to many laptop files at once?

The advanced disk software for the Tandy and Purple Computing 3½" disk drives.

When Traveling Software first introduced TS-DOS, a new standard was set for software operating the Purple Computing and Tandy portable 3½" disk drives.

Now, the standard is set even higher, with **TS-RANDOM,** an advanced disk operating system that gives you more power than ever before—with random access to many files at once!

Rated ★★★★ by "80-Micro Magazine"

TS-RANDOM is built into your computer's BASIC language and lets you write programs that retrieve information much faster than sequential-access methods. Up to eight files are supported for simultaneous access for input and output. A sample BASIC program is even included so you can modify according to your own applications.

And with **TS-RANDOM,** you can format your own disks for different size records, allowing up to 1600 per disk.

A new tagging feature too!

TS-RANDOM and TS-DOS come with a new tagging feature that lets you load, save, or kill several files at once—saving you time and energy. You can select specific files for tagging, or, tag all at once. This is ideal for loading or saving entire banks of memory on a file-by-file basis.

Recover disks and make quick copies with TS-RANDOM.

Special utilities for recovering erased or damaged disk files—and, performing quick disk copying—are also included with **TS-RANDOM.**

Available on a Plug-In ROM.

TS-RANDOM is available on disk as well as a plug-in ROM chip. The ROM version occupies very little RAM memory and includes both RAM recovery and Lapcoder TEXT compression options built in. Just one press of a key and any text file is compressed up to 50%—so you can store the equivalent of 150K of TEXT on a 100K disk. Or, 48K of TEXT in a 32K RAM bank.

F E A T U R E S

- **TS-RANDOM** allows simultaneous access to up to eight BASIC files at once.
- Support for multiple disk formats
- 400 to 1600 records per disk
- Includes a complete disk file recovery system and easy disk copying.
- New file tagging feature
- ROM version contains exclusive RAM recovery and TEXT compression.

P R I C I N G

TS-DOS (disk)	$69.95
TS-RANDOM (disk)	$89.95
TS-RANDOM (plug-in ROM)	$119.00

All TS-DOS owners can upgrade to a **TS-RANDOM** on disk for $20 or on ROM for $49.

To order ***TS-RANDOM*** *call toll free:*

1 800 343-8080

Or write: **Traveling Software**
North Creek Corporate Center
19310 North Creek Parkway
Bothell, Washington 98011
206-483-8088

TS-RANDOM

TRAVELING SOFTWARE

to program an EPROM on a ROM-STD or ROM-OPT module. Another is LL-100, a production circuit board that allows the Model 100 to control burning an EPROM through BASIC or machine language.

HOT STUFF

Burning a ROM means to imprint binary zeros onto an EPROM by applying carefully controlled voltages. EPROMs come from the factory with all binary bits preset in the one position. Radoyle's LL-100 changes certain ones to zeroes. If you want to erase the chip and start over, you can change all of the zeros back to ones by uncovering a small window at the top of the chip and exposing the chip to very bright ultraviolet light for several minutes.

The LL-100 is a simple, well-designed glass-epoxy circuit board. Though no dangerous voltages are present, the board is open to the environment — even a stray paper clip could cause trouble. But this shouldn't be a problem since specialized products like the LL-100 are generally used by sophisticated users in a test-bench or lab setting. If you're using one at home, though, keep the kids and dogs away.

The assumption underlying the LL-100 is that you know what you want on the chip. If you do a lot of machine-language programming, you may find it helpful to put data files or program files on the option ROM through the burning process. When the ROM is installed you can access the data under program control from the option ROM and load it into random-access memory (RAM), or use it in computations.

> The assumption is that you know what you want.

The LL-100 is not a product for the casual user. If you want to write your own machine-language routines or BASIC programs for the option ROM you will have to figure out how to run them from there. The user who is sophisticated enough to disassemble (turn binary signals into human-readable 80C85 assembler code) and understand the workings of Super ROM or Ultimate ROM II would presumably be able to figure out how to produce ROMable code. Once you've figured it out, the Radoyle equipment will let you burn the ROM and install it.

SOFTWARE INCLUDED

With the LL-100 you get a cassette full of useful programs for burning EPROMs, verifying that the burn was successful, copying the operating system into ROM and verifying that the copy was successful. There are machine-language routines for high-speed burning. All are well documented, making it easy for the knowledgeable user to adapt them for particular applications.

Ah, but what about the Tandy 102? The 102's operating-system ROM is

soldered rather than socketed in place. Any attempt to install a substitute ROM (regardless of whether the ROM-STD is used) would require substantial desoldering and the attendant risk of damage to the main PC board. Also, the option ROM socket in the 102 is oriented differently in its compartment so that the ROM-OPT cannot be used. Finally, the system bus connector on the 102 differs from that on the 100 so that the system bus cable of the LL-100 would require modification to fit on the 102.

Radoyle's documentation is written in a clear though necessarily technical style. Warnings are properly provided at several points reminding the user to be careful to back up important files and about the possibility of static electricity damaging the EPROMs.

SHOULD YOU DO IT?

The Model 100 family's operating system ROMs are copyrighted by Microsoft. If you copy part of the operating system into an EPROM, and add code of your own, and use it in a Model 100, are you infringing on that copyright?

The finely crafted fine print in the front of the Model 100 owner's manual forbids nearly any use of the ROM beyond turning the computer on. In particular the license language forbids disassembling the software, and disassembly would be a necessary part of any attempt to modify the operating system.

Do-it-Yourself EPROMS
Technology First/
Radoyle
9235 Chesapeake
Drive, Suite B
San Diego, CA 92123
LL-100 EPROM
Programmer $150
ROM-OPT $60
ROM-STD $60
TCJ-28 $50

The law is less than clear in cases like this, but common business sense suggests there wouldn't be a problem. The main concern is that the copyright holder, in this case Microsoft, not lose out of any deserved royalty. Anyone who fabricates such a modified operating system EPROM can only get value from it by plugging it into a Model 100. To do that he or she must have purchased the 100 and thus paid a royalty for use of the software. So it's not as though Microsoft would be losing any royalties that it would otherwise have earned.

But every situation differs. If you intend to market any ROM or other product that's based on copyright code — no matter whose — you should consult a lawyer knowledgeable in the complex and constantly changing field of software-related copyrights.

But don't let all that scare you; burning your own ROM containing assembler code that you've written yourself can add to the flexibility of your laptop computer — and can even save precious RAM. The sophisticated programmer and experimenter will enjoy the Radoyle line of EPROM equipment. It opens up a world of option ROM capability.

the load is complete, the file manager jumps back to a portion of DB.CO. This part of the utility undoes the entire patch, restores the original file manager and concludes by jumping to the start of the SCHEDL program.

Remember that you can have up to forty different file names on a single disk. With DB.CO you can have any of those data files loaded and displayed on your screen in seconds. The rapid-fire "load-and-go" feature of the utility will give you a near-database capability without the high cost of commercial software. All it takes is DB.CO to teach your disk drive and SCHEDL to work together for better portable computing.

—Michael Chandler

An Edge on Mastermind

Several years ago the game Mastermind became very popular. It was played this way: Player A selected four different colored pegs to be placed side by side on a board, hidden from Player B's view. Player B's task was to guess not only the color of each peg but also the order in which they were arranged.

Each time, player A gave player B a clue in the form of a red marker for each peg of the right color and in the right position; a white marker for each peg correctly colored but incorrectly placed; and no marker if the color and position were both wrong.
Player B's view. Player B's task was to guess not only the color of each peg but also the order in which they were arranged.

Each successive turn provided more clues, making it a little easier to deduce the answer. With good concentration, a player could usually come up with the correct sequence after six turns.

One of the drawbacks of the game was that it was quite easy for player A to unintentionally provide wrong clues — after all, it is human to err. This is where the Model 100 comes in. Substitute numbers for colors and let the computer become player A, and you are guaranteed that all the clues will be accurate.

The computer chooses a random four-digit sequence (lines 50 and 55) that can include leading zeros or contain two or more numbers that are the same. You enter your guesses when prompted and your selections are listed sequentially beginning at the top of the left column and continuing down to the bottom of the right column. The computer then places the clues next to your guesses on the right; a small black triangle, CHR$(167), indicates a correct number and position, and a white diamond, CHR$(157), indicates a correct number but wrong position. If you wish to change those characters, you may do so in lines 75 and 90.

You are allowed 14 tries, after which you are given the correct number and given the opportunity to play again. If you want to give up before the 14th try, enter four X's (upper case).

Lines 100-120 contain a routine for randomizing the relative positions of the clues so that you cannot tell from the order of the clues which numbers in any one guess are correct. This is an enjoyable way to develop those analytical powers — or just to pass the time while you're waiting for your next plane.

—Hamish Drummond

MSTRMD.BA: A game for the Model 100 or Tandy 200.

```
5 '              MSTRMD
10 '               by
15 '         Hamish Drummond
40 '
45 CLS
  :LINE (36, 0)-(36, 56)
  :LINE (101, 4)-(101, 52)
  :LINE (166, 0)-(166, 56)
  :LINE (36, 17)-(0, 17)
  :LINE (36, 36)-(0, 36)
  :LINE (166, 56)-(36, 56)
50 X=RND(-VAL(RIGHT$(TIME$, 2)))
  :Q=7
55 FOR A=1 TO 4
  :N$=MID$(STR$(INT(RND(1)*9)), 2)+N$
  :NEXT A
  :PRINT @40, CHR$(27)+"p"+"Choice"+CHR$(27)+"q"
60 L=L+1
  :N1$=N$
65 PRINT @80, ""
  :INPUT G$
  :PRINT @120, "       "
  :IF G$="XXXX" THEN 145 ELSE IF LEN(G$)<>4 OR
   ASC(G$)>57 OR ASC(G$)<48 THEN 65 ELSE G1$=G$
  :L1=L1+1
  :PRINT @Q+40*(L-1), G$
70 FOR A=1 TO 4
75 IF MID$(G1$, A, 1)=MID$(N1$, A, 1) THEN MID$
   (N1$, A, 1)="*"
  :MID$(G1$, A, 1)="/"
  :C$=C$+CHR$(167)
80 NEXT A
85 FOR A=1 TO 4
  :FOR B=1 TO 4
90 IF MID$(G1$, B, 1)=MID$(N1$, A, 1) THEN MID$
   (N1$, A, 1)="*"
  :MID$(G1$, B, 1)="/"
  :C$=C$+CHR$(157)
95 NEXT B, A
100 C$=C$+STRING$(4-LEN(C$), " ")
   :S$="1234"
105 C=INT(RND(1)*4+1)
110 IF MID$(S$, C, 1)="?" THEN 120
115 MID$(S$, C, 1)="?"
   :C1$=MID$(C$, C, 1)+C1$
120 IF S$<>"????" THEN 105
125 PRINT @(Q+5)+40*(L-1), C1$
   :C$=""
   :C1$=""
   :IF L=7 THEN L=0
   :Q=Q+11
130 IF L1=14 AND G$<>N$ THEN 145 ELSE IF L1=14
    THEN 140
135 IF G$<>N$ THEN 60
140 SOUND 4697, 5
   :SOUND 4184, 5
   :SOUND 3728, 5
   :SOUND 3516, 5
   :SOUND 3134, 5
   :SOUND 2793, 5
   :SOUND 2488, 5
   :SOUND 2348, 5
   :PRINT @108, "You guessed"
   :PRINT @148, "it in";L1
   :PRINT @188, "goes"
   :GOTO 150
145 SOUND 16383, 40
   :PRINT @68, "Nbr was ";N$
   :PRINT @150, "and you"
   :PRINT @228, "failed in";L1
150 PRINT @280, "Another go (Y/N)?";
   :R$=INKEY$
   :IF R$="" THEN 150 ELSE IF R$="Y" OR R$="y"
    THEN RUN ELSE MENU
```

6 ROM BANK

plus powerful Nicad battery pack that gives 30 hours of power to your Model 100

A joint PCSG/CRYPTRONICS project.

Now you can access LUCID, DISK+, WRITE ROM, RAM+ and others instantly.

The 6 ROM BANK plus battery pack lets you have the ultimate Model 100 system. When you get your unit just press in up to 6 ROMS into the sockets then push in a simple plug into your Model 100's ROM socket. The ROM bank has two posts on either end that insert into the two little holes on the underside of your Model 100.

The ROM bank props up the Model 100 at the same angle and height as those little legs you've seen. The ROM BANK itself is only about 1½" deep and it runs the width of your Model 100. It only weighs one pound. It not only installs instantly, but it pops free in a second if you need everything to lie flat in a briefcase.

Change from ROM to ROM with the touch of a thumb switch.

You can go from LUCID to WRITE to DISK+ to any other ROMS just by turning the thumb switch at the side of the ROM bank. The 6 ROM BANK is a sturdy well built construction that looks like it is a part of your Model 100.

What is also fantastic is that the ROM bank has a powerful NICAD battery and recharger built right in. This power source supplies up to 30 hours of life to your Model 100 with just 6 hours of recharge. What's nice is that it recharges right from your Model 100's power adapter. This is a quick charge system and if you need power in a hurry, you can get 6 hours of life for your Model 100 by just charging for an hour and half.

Suddenly, the Model 100 is a very powerful computer.

If you have the HOLMES portable disk drive with its powerful bundled database software package or the CRYPTRONICS 128K RAM expansion along with LUCID, WRITE ROM and DISK+ in the 6 ROM bank with its rapid rechargable NICAD power source, you have the ultimate portable system. It's all available only from PCSG.

As usual we don't want you to take our word for it. The 6 ROM BANK is sold on a 30 day trial. If you are not satisfied simply return within thirty days for a full refund. MC, VISA, AM. EX. or C.O.D.

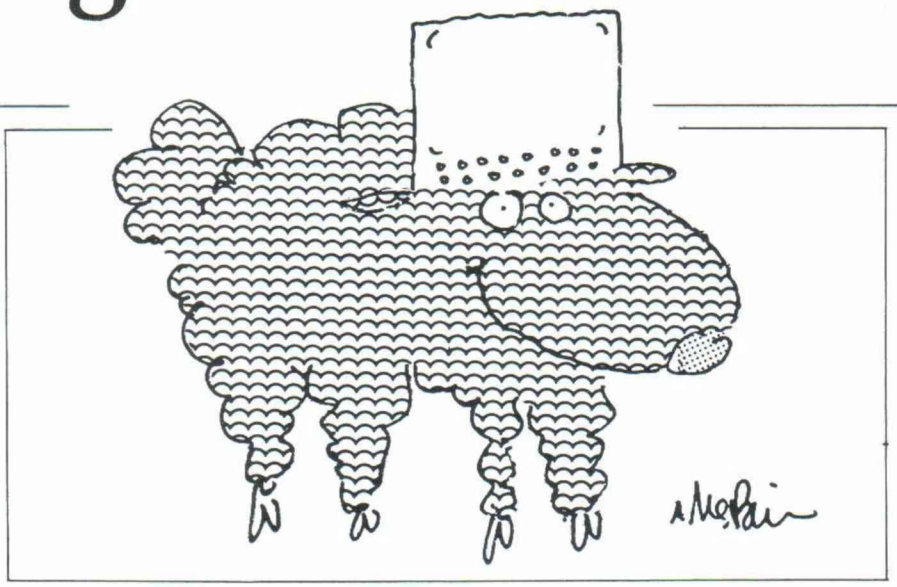
1-214-351-0564

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd. No. 207, Dallas, Tx. 75229

VISA / MASTER CARD / AM. EXP. / C.O.D.

Learning the Protocol



A communications protocol is nothing more than a way of talking that both parties have agreed to in advance. Perhaps the most universal telecom protocol is the ASCII character set. ASCII, which stands for American Standard Code for Information Interchange, is a way of encoding 128 alphanumeric characters and special codes in seven binary bits. ASCII says that the binary code for number zero is 0110001, for the letter A is 1000001 and for a typewriter carriage return is 0001101. We usually express ASCII codes in decimal (base 10) notation, so a carriage return is an ASCII 13. On the Model 100, ASCII codes are generated with the CHR$ and ASC functions.

ASCII isn't the only alphanumeric protocol in town. An earlier standard is EBCDIC, still used by large IBM mainframes. And ASCII has some interesting — and often frustrating — limitations. For example, some computers and printers assume that the ASCII 13 carriage return (which returns the print head to the left margin) implies a line feed. Other devices expect an explicit line feed character, ASCII 10. Similarly, some computer and printers don't recognize the form-feed code, ASCII 12, which should clear the display or advance to the next sheet of paper.

Another problem with the ASCII code set is that it defines a range of seven binary digits — but many computers, including the Model 100 use eight bits to define characters. That's a range of 256 possible codes, and ASCII covers only 128 of them. Each computer is free to define its own proprietary use for the "upper" character range.

The difference between seven and eight data bits reflects more than just the weakness in the ASCII system. That difference affects the lowest level of data communications error checking — parity.

During routine telecommunications, data is sent in groups of eight bits. If *parity* isn't used, the eight bits can describe the entire 256-character range mentioned above. But if parity checking is enabled, or used, the eighth bit takes on a special meaning.

There are four basic types of parity checking: *even, odd, ignore* and *none,* all relating to the number of zeros and ones in the ASCII character being transmitted. Take the earlier example: the character A is the binary 1000001. That's two ones and five zeros. If we're using even parity, the number of ones must be even. So, the transmitting computer adds a zero as a most-significant digit and sends 01000001. The computer that's receiving this transmission, which is already set for the *even parity protocol,* counts the number of binary ones. If it finds an odd number it knows that there has been an error in the transmission and takes appropriate action — on the Model 100, a checkboard-like character is displayed (ASCII 255). Similarly, if both computers were told to use odd parity, the character would have been transmitted as 11000001. Still, the actual information be transferred is only seven data bits — the left-most bit is meaningless except for error checking, and is always set back to zero by the receiving computer.

If the computers are set for parity *ignore* or *none,* the most significant binary digit is always zero. You're not gaining anything by using seven-bit transmission with no parity checking. Even parity is the most common — so use that setting whenever possible.

There is one exception: when it's important to transmit all eight binary bits. When you're transmitting a program or if your document has Code or Grph characters embedded in the text, you'll need to set the transmission protocols to eight data bits, with no parity checking.

So far we've covered three of the Model 100 *STAT* codes. The first character is an abbreviated transmission speed: *M* for 300 bits per second (bps) using the internal modem, or a digit for the RS-232 transmission speed. The second, either a seven or an eight, is the number of data bits. The third can be an *I, E, O* or *N,* depending on the parity selected. The fourth *STAT* parameter controls the number of *stop bits.* When a character is transmitted, it's always proceeded by a binary zero, called the *start bit.* It's also followed by one or two binary ones, the stop bits. The start and stop bits define the beginning and end of each character; a second stop bit gives the receiver a little more time to process each byte.

In almost every instance, you'll want to use only one stop bit. The only exceptions I can think of are when you're connected to a system that needs a little extra time between characters: 110-bps mechanical teletype machines are a good example. Also, some large mainframes expect their terminals to use two stop bits to aid in synchronization. As a rule, though, set that fourth *STAT* parameter to *1.*

—Alan L. Zeichick

SUPER ROM

Lucid Spreadsheet **Write ROM** **Database** **Outliner**

NOW YOU CAN REALLY HAVE IT ALL!

All on one ROM. Truly the finest four programs available for the Model 100 — guaranteed. Try it for 30 days. If you aren't blown away by the excellence return it for a full refund.

$199.95

The four best programs for the Model 100 all on one ROM. 32K of power without using any RAM for program storage. This is the PCSG Snap-In ROM that just presses easily into the little ROM socket in the compartment on the back. You access the four right from the main menu like built-ins.

Write ROM — the definitive word processor for the Model 100. Function key formatting or dot commands. Search and replace. Library feature — inserts words, phrases or whole documents into text from just a code. MAP lets you see a picture of your document. In all there are 60 features and functions. No one can claim faster operation. FORM lets you create interactive forms with on-screen prompts that you can answer from the keyboard. Nothing else for the Model 100 compares with the features of Write ROM. Exactly the same as the Write ROM sold as a single program. Infoworld says it "makes the Model 100 a viable writing unit ... surpassed our highest expectations for quality and clarity."

Lucid Spreadsheet: This is the one PICO magazine says "blows Multiplan right out of the socket" and Infoworld performance rated as "excellent" and said "makes the Model 100 compute." Gives you features you cannot get with Lotus 123. Lets you build spreadsheets in your Model 100 that would consume 140-150K on a desktop. Program generating capability with no programming knowledge required. Variable column widths. Includes find and sort with function key control. It's fast, recalculates like lightning. No feature has been taken from the original, only new ones added.

Database: This is a relational data base like no other. You can do everything from mailing lists to invoices. No complicated pseudo-coding, you create input screens as simply as typing into TEXT. You are not limited by size; you can have as large an input screen as you wish. Prints out reports or forms, getting information from as many files as you like. Complete math between fields. Total interface with Lucid worksheets.

Outliner: Does everything that Thinktank does on a PC but a whole lot better. Includes a Sort for your headlines. Lets you have headlines of up to 240 characters. Has cloning, hoisting and sideways scroll up to 250 characters. Like Lucid, this one sets a new standard for outliners. This is the way to plan and organize your projects.

Present Lucid and Write ROM owners can upgrade for $100. If you have both it's $75.

As usual PCSG sells the Super ROM on a thirty day guarantee. If for any reason you are not satisfied, simply return it for a full refund.

We are excited about this product. Super ROM gives the Model 100 the true power of a desktop. No other multi-program ROM has software that compares. But don't take our word for it. We invite you to make that comparison yourself. Priced at $199.95 on Snap-In ROM.

Got stuck with somebody else's multi-ROM? We'll upgrade it for $100.

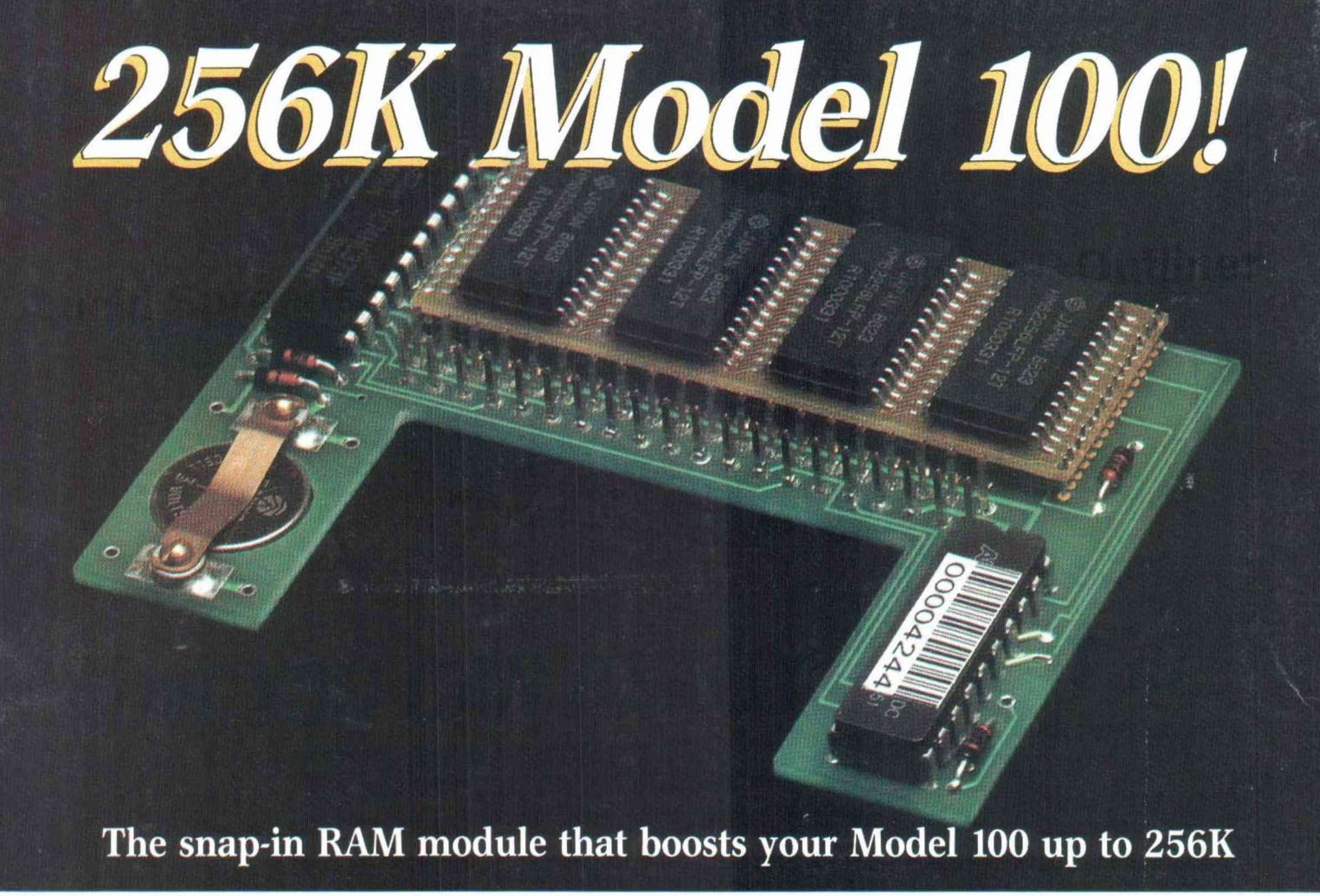
(214) 351-0564

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd., #206, Dallas, TX 75229

MC, Visa, American Express, Check, or C.O.D.

256K Model 100!



The snap-in RAM module that boosts your Model 100 up to 256K

Circle 470 on Reader Service card.

Here's how we did it.

We packaged seven 32K RAM banks (224K) on a neat little module that snaps into the rear expansion port of your Model 100. Now you can have as much RAM memory in one Model 100 as eight 32K Model 100s.

Versatile Memory

Each of our additional 32K RAM banks has it's own command of the software that comes standard in your Model 100. All of the memory banks can be easily and directly accessed from any of the other banks. Transfer your files from bank to bank simply by using the function keys. We'll even display on the menu screen how many bytes any file is when you move the menu cursor over the file name.

You can really use more memory!

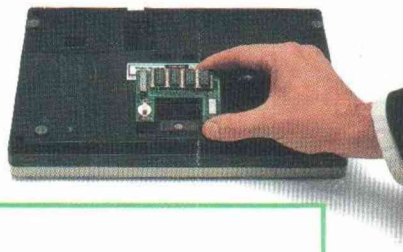
Most of us have run out of RAM memory at one time or another in our Model 100. Imagine how nice it would be to have so much more memory in just one Model 100. Salesmen, journalists, scientists, any Model 100 user can maintain databases in some banks while dedicating other banks to sales letters, spreadsheets or other programs. For all of you the benefit is obvious—you carry more data with you in one small reliable package. (Memory entered in all of the additional RAM banks are backed up by a six year lithium power cell.)

Installation's a snap!

The RAM module simply snaps into the rear expansion compartment of your Model 100 and you're ready to go. There is no programming required. (The PG Design RAM module expands the RAM memory of any Model 100.)

Order Today

PG Design manufactures many fine memory expansion products for Tandy portable computers. All are guaranteed to work as promised or your money back. All are built to last.



Order Today

64K modules—$300
128K modules—$425
224K modules—$575
(M100) 8K modules—$24.95 ea./3 for $69
*available for NEC 8201A
(M102) 8K modules—$10.95
(M200) 24K modules—$75 ea./2 or more, $70

We accept Visa, Master Card, personal checks and money orders. We ship within five working days of receiving your order.

*Tandy is a trademark of Tandy Corp.

Dealers inquire

PG DESIGN ELECTRONICS, INC.
37560 Thirty-one Mile Road, Richmond, MI 48062

Call 313/727-2744